Deep Stochastic Mechanics

Elena Orlova [1] **Aleksei Ustimenko** [2] **Ruoxi Jiang** [1] **Peter Y. Lu** [3] **Rebecca Willett** [1 4]

Abstract

This paper introduces a novel deep-learning-based approach for numerical simulation of a time-evolving Schrödinger equation inspired by stochastic mechanics and generative diffusion models. Unlike existing approaches, which exhibit computational complexity that scales exponentially in the problem dimension, our method allows us to adapt to the latent low-dimensional structure of the wave function by sampling from the Markovian diffusion. Depending on the latent dimension, our method may have far lower computational complexity in higher dimensions. Moreover, we propose novel equations for stochastic quantum mechanics, resulting in quadratic computational complexity with respect to the number of dimensions. Numerical simulations verify our theoretical findings and show a significant advantage of our method compared to other deep-learning-based approaches used for quantum mechanics.

1. Introduction

Mathematical models for many problems in nature appear in the form of partial differential equations (PDEs) in high dimensions. Given access to precise solutions of the many-electron time-dependent Schrödinger equation (TDSE), a vast body of scientific problems could be addressed, including in quantum chemistry (Cances et al., 2003; Nakatsuji, 2012), drug discovery (Ganesan et al., 2017; Heifetz, 2020), condensed matter physics (Boghosian & Taylor IV, 1998; Liu et al., 2013), and quantum computing (Grover, 2001; Papageorgiou & Traub, 2013). However, solving high-dimensional PDEs and the Schrödinger equation, in particular, are notoriously difficult problems in scientific computing due to the well-known curse of dimensionality: the computational complexity grows exponentially as a function of the dimensionality of the problem (Bellman, 2010). Traditional numerical solvers have been limited to dealing with problems in rather low dimensions since they rely on a grid.

Deep learning is a promising way to avoid the curse of dimensionality (Poggio et al., 2017; Madala et al., 2023). However, no known deep learning approach avoids it in the context of the TDSE (Manzhos, 2020). Although generic deep learning approaches have been applied to solving the TDSE (E & Yu, 2017; Han et al., 2018; Raissi et al., 2019; Weinan et al., 2021), this paper shows that it is possible to get performance improvements by developing an approach specific to the TDSE by incorporating quantum physical structure into the deep learning algorithm itself.

We propose a method that relies on a stochastic interpretation of quantum mechanics (Nelson, 1966; Guerra, 1995; Nelson, 2005) and is inspired by the success of deep diffusion models that can model complex multi-dimensional distributions effectively (Yang et al., 2022); we call it *Deep Stochastic Mechanics (DSM)*. Our approach is not limited to only the linear Schrödinger equation but can be adapted to Klein-Gordon, Dirac equations (Serva, 1988; Lindgren & Liukkonen, 2019), and to the non-linear Schrödinger equations of condensed matter physics, e.g., by using mean-field stochastic differential equations (SDEs) (Eriksen, 2020), or McKean-Vlasov SDEs (dos Reis et al., 2022).

1.1. Problem formulation

The Schrödinger equation, a governing equation in quantum mechanics, predicts the future behavior of a dynamic system for $0 \leq t \leq T$ and $\forall x \in \mathcal{M}$:

$$i\hbar\partial_t\psi(x,t) = \mathcal{H}\psi(x,t), \tag{1}$$

$$\psi(x,0) = \psi_0(x), \tag{2}$$

where $\psi : \mathcal{M} \times [0,T] \to \mathbb{C}$ is a wave function defined over a manifold $\mathcal{M}$, and $\mathcal{H}$ is a self-adjoint operator acting on a Hilbert space of wave functions. For simplicity of future derivations, we consider a case of a spinless particle[1] in

[1]Department of Computer Science, University of Chicago, Chicago, USA [2]ShareChat, London, UK [3]Department of Physics, University of Chicago, Chicago, USA [4]Department of Statistics, University of Chicago, Chicago, USA. Correspondence to: Aleksei Ustimenko <research@aleksei.uk>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

[1]A multi-particle case is covered by considering $d = 3n$, where n – the number of particles.

$\mathcal{M} = \mathbb{R}^d$ moving in a smooth potential $V : \mathbb{R}^d \times [0, T] \to \mathbb{R}_+$. In this case, $\mathcal{H} = -\frac{\hbar^2}{2}\mathrm{Tr}(m^{-1}\nabla^2) + V$, where $m \in \mathbb{R}^d \otimes \mathbb{R}^d$ is a mass tensor. The probability density of finding a particle at position x is $|\psi(x, t)|^2$. A notation list is given in Appendix A.

Given initial conditions in the form of samples drawn from density $\psi_0(x)$, we wish to draw samples from $|\psi(x, t)|^2$ for $t \in (0, T]$ using a neural-network-based approach that can adapt to latent low-dimensional structures in the system and sidestep the curse of dimensionality. Rather than explicitly estimating $\psi(x, t)$ and sampling from the corresponding density, we devise a strategy that directly samples from an approximation of $|\psi(x, t)|^2$, concentrating computation in high-density regions. When regions where the density $|\psi(x, t)|^2$ lie in a latent low-dimensional space, our sampling strategy concentrates computation in that space, leading to the favorable scaling properties of our approach.

2. Related Work

Physics-Informed Neural Networks (PINNs) (Raissi et al., 2019) are general-purpose tools that are widely studied for their ability to solve PDEs and can be applied to solve Equation (1). However, this method is prone to the same issues as classical numerical algorithms since it relies on a collection of collocation points uniformly sampled over the domain $\mathcal{M} \subseteq \mathbb{R}^d$. In the remainder of the paper, we refer to this as a 'grid' for simplicity of exposition. Another recent paper by Bruna et al. (2022) introduces Neural Galerkin schemes based on deep learning, which leverage active learning to generate training data samples for numerically solving real-valued PDEs. Unlike collocation-points-based methods, this approach allows theoretically adaptive data collection guided by the dynamics of the equations if we could sample from the wave function effectively.

Another family of approaches including DeepWF (Han et al., 2019), FermiNet (Pfau et al., 2020), and PauliNet (Hermann et al., 2020) reformulates the problem (1) as maximization of an energy functional that depends on the solution of the stationary Schrödinger equation. This approach sidesteps the curse of dimensionality but cannot be applied to the time-dependent wave function setting considered in this paper.

The only thing that one can experimentally obtain is samples from the quantum mechanics density. So, it makes sense to focus on obtaining samples from the density rather than attempting to solve the Schrödinger equation; these samples can be used to predict the system's behavior without conducting real-world experiments. Based on this observation, there are a variety of quantum Monte Carlo (MC) methods (Barker, 1979; Corney & Drummond, 2004; Austin et al., 2012), which rely on estimating expectations of observables rather than the wave function itself, resulting in improved computational efficiency. However, these methods still encounter the curse of dimensionality due to recovering the full-density operator. The density operator in atomic simulations is concentrated on a lower dimensional manifold of such operators (Eriksen, 2020), suggesting that methods that adapt to this manifold can be more effective than high-dimensional grid-based methods. Deep learning has the ability to adapt to this structure. Numerous works explore the time-dependent Variational Monte Carlo (t-VMC) schemes (Carleo et al., 2017; Carleo & Troyer, 2017; Schmitt & Heyl, 2020; Yao et al., 2021) for simulating many-body quantum systems. Their applicability is often tailored to a specific problem setting as these methods require significant prior knowledge to choose a good variational ansatz function. As highlighted by Sinibaldi et al. (2023), t-VMC methods may encounter challenges related to systematic statistical bias or exponential sample complexity, particularly when the wave function contains zeros.

As noted in Schlick (2010), knowledge of the density is unnecessary for sampling. We need a score function $\nabla \log \rho$ to be able to sample from it. The fast-growing field of generative modeling with diffusion processes demonstrates that for high-dimensional densities with low-dimensional manifold structure, it is incomparably more effective to learn a score function than the density itself (Ho et al., 2020; Yang et al., 2022).

For high-dimensional real-valued PDEs, there exist a variety of classic and deep learning-based approaches that rely on sampling from diffusion processes, e.g., Cliffe et al. (2011); Warin (2018); Han et al. (2018); Weinan et al. (2021). Those works rely on the Feynman-Kac formula (Del Moral, 2004) to obtain an estimator for the solution to the PDE. However, for the Schrödinger equation, we need an analytical continuation of the Feynman-Kac formula on an imaginary time axis (Yan, 1994) as it is a complex-valued equation. This requirement limits the applicability of this approach to our setting. BSDE methods studied by Nüsken & Richter (2021b;a) are closely related to our approach, but they are developed for the elliptic version of the Hamilton–Jacobi–Bellman (HJB) equation. We consider the hyperbolic HJB setting, for which the existing method cannot be applied.

3. Contributions

We are inspired by works of Nelson (1966; 2005), who has developed a stochastic interpretation of quantum mechanics, so-called stochastic mechanics, based on a Markovian diffusion. Instead of solving the Schrödinger equation(1), our method aims to learn the stochastic mechanical process's osmotic and current velocities equivalent to classical quantum mechanics. Our formulation differs from the original one (Nelson, 1966; Guerra, 1995; Nelson, 2005), as we derive

equivalent differential equations describing the velocities that do not require the computation of the Laplacian operator. Another difference is that our formulation interpolates anywhere between stochastic mechanics and deterministic Pilot-wave theory (Bohm, 1952). More details are given in Appendix E.4.

We highlight the main contributions of this work as follows:

- We propose to use a stochastic formulation of quantum mechanics (Nelson, 1966; Guerra, 1995; Nelson, 2005) to create an efficient and theoretically sound computational framework for quantum mechanics simulation. We accomplish our result by using stochastic mechanics equations stemming from Nelson's formulation. In contrast to Nelson's original expressions, which rely on second-order derivatives like the Lagrangian, our expressions rely solely on first-order derivatives – specifically, the gradient of the divergence operator. This formulation, which is more amenable to neural network-based solvers, results in a reduction in the computational complexity of the loss evaluation from cubic to quadratic in dimension.
- We prove theoretically in Section 4.3 that the proposed loss function upper bounds the L_2 distance between the approximate process and the 'true' process that samples from the quantum density, which implies that if loss converges to zero, then the approximate process strongly converges to the 'true' process. Our theoretical finding offers a simple mechanism for guaranteeing the accuracy of our predicted solution, even in settings in which no baseline methods are computationally tractable.
- We empirically estimate the performance of our method in various settings. Our approach shows a superior advantage to PINNs and t-VMC in terms of accuracy. We also conduct an experiment for non-interacting bosons where our method reveals linear convergence time in the dimension, operating easily in a higher-dimensional setting. Another interacting bosons experiment highlights the favorable scaling properties of our approach in terms of memory and computing time compared to a grid-based numerical solver. While our theoretical analysis establishes an $\mathcal{O}(d^2)$ bound on the algorithmic complexity, we observe an empirical scaling closer to $\mathcal{O}(d)$ for the memory and compute requirements as the problem dimension d increases due to parallelization in modern machine learning frameworks.

Table 1 compares properties of methods for solving Equation (1). For numerical solvers, the number of grid points scales as $\mathcal{O}(N^{\frac{d}{2}+1})$ as N is the number of discretization points in time, and $\sqrt{N}$ is the number of discretization points in each spatial dimension. We assume a numerical solver aims for a precision $\varepsilon = \mathcal{O}(\frac{1}{\sqrt{N}})$. In the context of neural networks, the iteration complexity is dominated by loss evaluation. For PINNs, N_f denotes the number of collocation points used to enforce physics-informed constraints in the spatio-temporal domain for $d = 1$. The original PINN formulation faces an exponential growth in the number of collocation points with respect to the problem dimension, $\mathcal{O}(N_f^d)$, posing a significant challenge in higher dimensions. Subsampling $\mathcal{O}(d)$ collocation points in a non-adaptive way leads to poor performance for high-dimensional problems.

For both t-VMC and FermiNet, H_d denotes the number of MC iterations required to draw a single sample. The t-VMC approach requires calculating a matrix inverse, which generally exhibits a cubic computational complexity of $\mathcal{O}(d^3)$ and may suffer from numerical instabilities. Similarly, the FermiNet method, which is used for solving the time-independent Schrödinger equation to find ground states, necessitates estimating matrix determinants, an operation that also scales as $\mathcal{O}(d^3)$. We note that for our DSM approach, N is independent of d. We focus on lower bounds on iteration complexity and known bounds for the convergence of non-convex stochastic gradient descent (Fehrman et al., 2019) that scales polynomial with ε^{-1}.

4. Deep Stochastic Mechanics

There is a family of diffusion processes that are equivalent to Equation (1) in a sense that all time-marginals of any such process coincide with $|\psi(x,t)|^2$; we refer to Appendix E for derivation. Assuming $\psi(x,t) = \sqrt{\rho(x,t)}e^{iS(x,t)}$, we define:

$$\begin{aligned} v(x,t) &= \frac{\hbar}{m}\nabla S(x,t), \\ u(x,t) &= \frac{\hbar}{2m}\nabla \log \rho(x,t). \end{aligned} \tag{3}$$

Our method relies on the following stochastic process with $\nu \geq 0$ [2], which corresponds to sampling from $\rho = \left|\psi(x,t)\right|^2$ (Nelson, 1966):

$$\begin{aligned} &\mathrm{d}Y(t) = (v(Y(t),t) + \nu u(Y(t),t))\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\vec{W}, \\ &Y(0) \sim \left|\psi_0\right|^2, \end{aligned} \tag{4}$$

where u is an osmotic velocity, v is a current velocity and $\vec{W}$ is a standard (forward) Wiener process. Process $Y(t)$ is called the Nelsonian process. Since we don't know the true u, v, we instead aim at approximating them with the process

[2] $\nu = 0$ is allowed if and only if ψ_0 is sufficiently regular, e.g., $|\psi_0|^2 > 0$ everywhere.

Table 1. Comparison of different approaches for simulating quantum mechanics.

METHOD	DOMAIN	TIME EVOLVING	ADAPTIVE	ITERATION COMPLEXITY	OVERALL COMPLEXITY
PINN (RAISSI ET AL., 2019)	COMPACT	✓	✗	$\mathcal{O}(N_f^d)$	$\geq \mathcal{O}(N_f^d \text{poly}(\varepsilon^{-1}))$
FERMINET (PFAU ET AL., 2020)	$\mathbb{R}^d$	✗	✓	$\mathcal{O}(H_d d^3)$	$\geq \mathcal{O}(H_d d^3 \text{poly}(\varepsilon^{-1}))$
T-VMC	$\mathbb{R}^d$	✓	✓	$\mathcal{O}(H_d d^3)$	$\geq \mathcal{O}(H_d d^3 \text{poly}(\varepsilon^{-1}))$
NUM. SOLVER	COMPACT	✓	✗	N/A	$\mathcal{O}(d\varepsilon^{-d-2})$
DSM (OURS)	$\mathbb{R}^d$	✓	✓	$\mathcal{O}(Nd^2)$	$\geq \mathcal{O}(Nd^2 \text{poly}(\varepsilon^{-1}))$

defined using neural network approximations v_θ, u_θ:

$$\mathrm{d}X(t) = (v_\theta(X(t),t) + \nu u_\theta(X(t),t))\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\vec{W}, \quad X(0) \sim |\psi_0|^2. \tag{5}$$

Any numerical integrator can be used to obtain samples from the diffusion process. The simplest one is the Euler–Maruyama integrator (Kloeden & Platen, 1992):

$$X_{i+1} = X_i + (v_\theta(X_i,t_i) + \nu u_\theta(X_i,t_i))\epsilon + \mathcal{N}\big(0, \frac{\nu\hbar}{m}\epsilon I_d\big), \tag{6}$$

where $\epsilon > 0$ denotes a step size, $0 \leq i < \frac{T}{\epsilon}$, and $\mathcal{N}(0, I_d)$ is a Gaussian distribution. We consider this integrator in our work. Switching to higher-order integrators, e.g., the Runge-Kutta family of integrators (Kloeden & Platen, 1992), can potentially enhance efficiency and stability when ϵ is larger.

The diffusion process from Equation (4) achieves sampling from $\rho = |\psi(x,t)|^2$ for each $t \in [0,T]$ for known u and v. Assume that $\psi_0(x) = \sqrt{\rho_0(x)}e^{iS_0(x)}$. Our approach relies on the following equations for the velocities:

$$\partial_t v = -\frac{1}{m}\nabla V + \langle u, \nabla\rangle u - \langle v, \nabla\rangle v + \frac{\hbar}{2m}\nabla\langle\nabla, u\rangle, \tag{7a}$$

$$\partial_t u = -\nabla\langle v, u\rangle - \frac{\hbar}{2m}\nabla\langle\nabla, v\rangle, \tag{7b}$$

$$v_0(x) = \frac{\hbar}{m}\nabla S_0(x), \; u_0(x) = \frac{\hbar}{2m}\nabla \log\rho_0(x). \tag{7c}$$

These equations are derived in Appendix E.1 and are equivalent to the Schrödinger equation. As mentioned, our equations differ from the canonical ones developed in Nelson (1966); Guerra (1995). In particular, the original formulation from Equation (26), which we call the *Nelsonian version*, includes the Laplacian of u; in contrast, *our version* in (7a) uses the gradient of the divergence operator. These versions are equivalent in our setting, but our version has significant computational advantages, as we describe later in Remark 4.1.

4.1. Learning Drifts

This section describes how we learn the velocities $u_\theta(X,t)$ and $v_\theta(X,t)$, parameterized by neural networks with parameters θ. We propose to use a combination of three losses: two of them come from the Navier-Stokes-like equations (7a), (7b), and the third one enforces the initial conditions (7c). We define non-linear differential operators that appear in Equation (7a), (7b):

$$\mathcal{D}_u[v,u,x,t] = -\nabla\langle v(x,t), u(x,t)\rangle - \frac{\hbar}{2m}\nabla\langle\nabla, v(x,t)\rangle, \tag{8}$$

$$\mathcal{D}_v[v,u,x,t] = -\frac{1}{m}\nabla V(x,t) + \frac{1}{2}\nabla\|u(x,t)\|^2 - \frac{1}{2}\nabla\|v(x,t)\|^2 + \frac{\hbar}{2m}\nabla\langle\nabla, u(x,t)\rangle. \tag{9}$$

We aim to minimize the following losses:

$$L_1(v_\theta,u_\theta) = \int_0^T \mathbb{E}^X \big\|\partial_t u_\theta(X(t),t) - \mathcal{D}_u[v_\theta,u_\theta,X(t),t]\big\|^2 \mathrm{d}t, \tag{10}$$

$$L_2(v_\theta,u_\theta) = \int_0^T \mathbb{E}^X \big\|\partial_t v_\theta(X(t),t) - \mathcal{D}_v[v_\theta,u_\theta,X(t),t]\big\|^2 \mathrm{d}t, \tag{11}$$

$$L_3(v_\theta,u_\theta) = \mathbb{E}^X\|u_\theta(X(0),0) - u_0(X(0))\|^2, \tag{12}$$

$$L_4(v_\theta,u_\theta) = \mathbb{E}^X\|v_\theta(X(0),0) - v_0(X(0))\|^2, \tag{13}$$

where u_0, v_0 are defined in Equation (7c). Finally, we define a combined loss using a weighted sum with $w_i > 0$:

$$\mathcal{L}(\theta) = \sum_{i=1}^4 w_i L_i(v_\theta,u_\theta). \tag{14}$$

The basic idea of our approach is to sample new trajectories using Equation (6) with $\nu = 1$ for each iteration τ. These trajectories are then used to compute stochastic estimates of the loss from Equation (14), and then we back-propagate

gradients of the loss to update θ. We re-use recently generated trajectories to reduce computational overhead as SDE integration cannot be paralleled. The training procedure is summarized in Algorithm 1 and Figure 1; a more detailed version is given in Appendix B.

Algorithm 1 Training algorithm pseudocode

Input ψ_0 – initial wave-function, M – epoch number, B – batch size, other parameters (optimizer parameters, physical constants, Euler–Maruyama parameters; see Appendix B)
Initialize NNs $u_{\theta_0}, v_{\theta_0}$
for each iteration $0 \leq \tau < M$ **do**
  Sample B trajectories using $u_{\theta_\tau}, v_{\theta_\tau}$ via Equation (6) with $\nu = 1$
  Estimate loss $\mathcal{L}(v_{\theta_\tau}, u_{\theta_\tau})$ from Equation (14) over the sampled trajectories
  Back-propagate gradients to get $\nabla_\theta \mathcal{L}(v_{\theta_\tau}, u_{\theta_\tau})$
  An optimizer step to get $\theta_{\tau+1}$
end for
output $u_{\theta_M}, v_{\theta_M}$

We use trained $u_{\theta_M}, v_{\theta_M}$ to simulate the forward diffusion for $\nu \geq 0$ given $X_0 \sim \mathcal{N}(0, I_d)$:

$$X_{i+1} = X_i + (v_{\theta_M}(X_i, t_i) + \nu u_{\theta_M}(X_i, t_i))\epsilon + \mathcal{N}\big(0, \frac{\hbar}{m}\nu\epsilon I_d\big). \quad (15)$$

Appendix G describes a wide variety of possible ways to apply our approach for estimating an arbitrary quantum observable, singular initial conditions like $\psi_0 = \delta_{x_0}$, singular potentials, correct estimations of observable that involve measurement process, and recovering the wave function from u, v.

Although PINNs can be used to solve Equations (7a), (7b), that approach would suffer from having fixed sampled density (see Section 5). Our method, much like PINNs, seeks to minimize the residuals of the PDEs from Equations (7a) and (7b). However, we do so on the distribution generated by sampled trajectories $X(t)$, which in turn depends on current neural approximations v_θ, u_θ. This allows our method to focus only on high-density regions and alleviates the inherent curse of dimensionality that comes from reliance on a grid.

4.2. Algorithmic Complexity

Our formulation of stochastic mechanics with novel Equations (7) is much more amenable to automatic differentiation tools than if we developed a neural diffusion approach based on the Nelsonian version. In particular, the original formulation uses the Laplacian operator Δu that naively requires $\mathcal{O}(d^3)$ operations, which might become a major bottleneck for scaling them to many-particle systems. While a stochastic trace estimator (Hutchinson, 1989) may seem an option to reduce the computational complexity of Laplacian calculation to $\mathcal{O}(d^2)$, it introduces a noise of an amplitude $\mathcal{O}(\sqrt{d})$. Consequently, a larger batch size (as $\mathcal{O}(d)$) is necessary to offset this noise resulting in still a cubic complexity.

Remark 4.1. The algorithmic complexity w.r.t. d of computing differential operators from Equations (8), (9) for velocities u, v is $\mathcal{O}(d^2)$. [3]

This remark is proved in Appendix E.5. This trick with the gradient of divergence can be used as we rely on the fact that the velocities u, v are full gradients, which is not the case for the wave function $\psi(x, t)$ itself.

We expect that one of the factors of d associated with evaluating a d-dimensional function gets parallelized over in modern machine learning frameworks, so we can see a linear scaling even though we are using an $\mathcal{O}(d^2)$ method. We will see such behavior in our experiments.

4.3. Theoretical Guarantees

To further justify the effectiveness of our loss function, we prove the following theorem in Appendix F:

Theorem 4.2. *(Strong Convergence Bound) We have the following bound between processes Y (the Nelsonian process that samples from $|\psi|^2$) and X (the neural approximation with v_θ, u_θ):*

$$\sup_{t \leq T} \mathbb{E}\|X(t) - Y(t)\|^2 \leq C_T \mathcal{L}(v_\theta, u_\theta), \quad (16)$$

where constant C_T is defined explicitly in F.13.

This theorem means optimizing the loss leads to a strong convergence of the neural process X to the Nelsonian process Y, and that the loss value directly translates into an improvement of L_2 error between the processes. The constant C depends on a horizon T and Lipshitz constants of u, v, u_θ, v_θ. It also hints that we have a 'low-dimensional' structure when Lipshitz constants of u, v, u_θ, v_θ are $\ll d$, which is the case of low-energy regimes (as large Lipshitz smoothness constant implies large value of the Laplacian and, hence, energy) and with the proper selection of a neural architecture (Aziznejad et al., 2020).

5. Experiments

Experimental setup As a baseline, we use an analytical or numerical solution. We compare our method's (DSM) performance with PINNs and t-VMC. In the case of non-interacting particles, the models are feed-forward neural networks with one hidden layer and a hyperbolic tangent

[3] Estimation of a term $\nabla V(x, t)$ might have different computational complexity from $\mathcal{O}(d)$, $\mathcal{O}(d^2)$, or even higher depending on a particle interaction type.

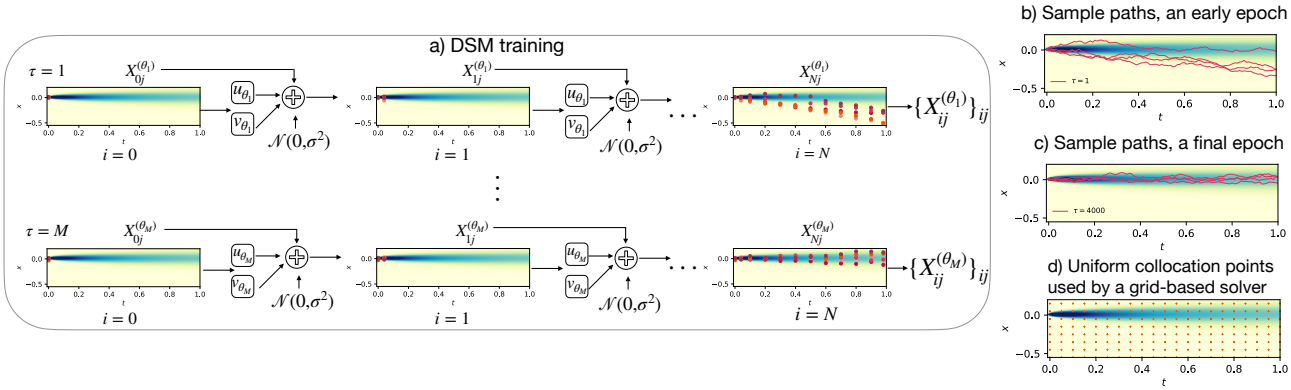


Figure 1. An illustration of our approach. Blue regions in the plots correspond to higher-density regions. (a) DSM training scheme: at every epoch τ, we generate B full trajectories $\{X_{ij}\}_{ij}$, $i = 0, ..., N$, $j = 1, ..., B$. Then, we update the weights of our NNs. (b) An illustration of sampled trajectories at the early epoch. (c) An illustration of sampled trajectories at the final epoch. (d) Collocation points for a grid-based solver where it should predict values of $\psi(x, t)$.

(tanh) activation function. We use a similar architecture with residual connection blocks and a tanh activation function when studying interacting particles. Further details on numerical solvers, architecture, training procedures, hyperparameters of our approach, PINNs, and t-VMC can be found in Appendix C. Additional experiment results are given in Appendix D. The code of our experiments can be found on GitHub [4]. We only consider bosonic systems, leaving fermionic systems for further research.

Evaluation metrics We estimate errors between true and predicted values of the mean and the variance of a coordinate X_i at time $i = 1, \dots, T$ as the relative L_2-norm, namely $\mathcal{E}_m(X_i)$ and $\mathcal{E}_v(X_i)$. The standard deviation (confidence intervals) of the observables are indicated in the results. True v and u values are estimated numerically with the finite difference method. Our trained u_θ and v_θ should output these values. We measure errors $\mathcal{E}(u)$ and $\mathcal{E}(v)$ as the L_2-norm between the true and predicted values in $L_2(\mathbb{R}^d \times [0, T], \mu)$ with $\mu(\mathrm{d}x, \mathrm{d}t) = |\psi(x, t)|^2 \mathrm{d}x\mathrm{d}t$.

5.1. Non-interacting Case: Harmonic Oscillator

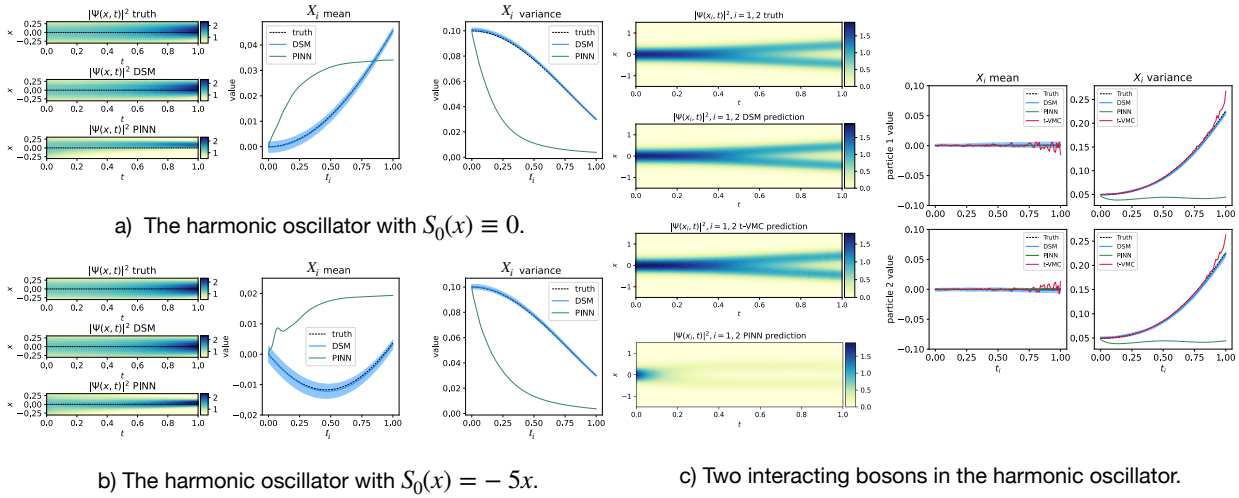
We consider a harmonic oscillator model with $x \in \mathbb{R}^1$, $V(x) = \frac{1}{2}m\omega^2(x - 0.1)^2$, $t \in [0, 1]$ and where $m = 1$ and $\omega = 1$. The initial wave function is given as $\psi(x, 0) \propto e^{-x^2/(4\sigma^2)}$. Then $u_0(x) = -\frac{\hbar x}{2m\sigma^2}$, $v_0(x) \equiv 0$. $X(0)$ comes from $X(0) \sim \mathcal{N}(0, \sigma^2)$, where $\sigma^2 = 0.1$.

We use the numerical solution as the ground truth. Our approach is compared with a PINN. The PINN input data consists of $N_0 = 1000$ points sampled for estimating $\psi(x, 0)$, $N_b = 300$ points for enforcing the boundary conditions (we assume zero boundary conditions), and $N_f = 60000$ collocation points to enforce the corresponding equation inside the solution domain, all points sampled uniformly for $x \in [-2, 2]$ and $t \in [0, 1]$.

Figure 2(a) summarizes the results of our experiment. The left panel of the figure illustrates the evolution of the density $|\psi(x, t)|^2$ over time for different methods. It is evident that our approach accurately captures the density evolution, while the PINN model initially aligns with the ground truth but deviates from it over time. Sampling collocation points uniformly when density is concentrated in a small region explains why PINN struggles to learn the dynamics of Equation (1); we illustrate this effect in Figure 1 (d). The right panel demonstrates observables of the system, the averaged mean of X_i, and the averaged variance of X_i. Our approach consistently follows the corresponding distribution of X_i. On the contrary, the predictions of the PINN model only match the distribution at the initial time steps but fail to accurately represent it as time elapses. Table 2 shows the error rates for our method and PINNs. In particular, our method performs better in terms of all error rates than the PINN. These findings emphasize the better performance of the proposed method in capturing the dynamics of the Schrödinger equation compared to the PINN model.

We also consider a non-zero initial phase $S_0(x) = -5x$. It corresponds to the initial impulse of a particle. Then $v_0(x) \equiv -\frac{5\hbar}{m}$. The PINN inputs are $N_0 = 3000$, $N_b = 300$ points, and $N_f = 80000$ collocation points. Figure 2 (b) and Table 2 present the results of our experiment. Our method consistently follows the corresponding ground truth, while the PINN model fails to do so. It indicates the ability of our method to accurately model the behavior of the quantum system.

[4] https://github.com/elena-orlova/deep-stochastic-mechanics



a) The harmonic oscillator with $S_0(x) \equiv 0$.

b) The harmonic oscillator with $S_0(x) = -5x$.

c) Two interacting bosons in the harmonic oscillator.

Figure 2. Simulation results of PINN and our DSM method: (a) and (b) correspond to a particle in the harmonic oscillator with different initial phases; (c) corresponds to two interacting bosons in the harmonic oscillator. The left panel of these figures corresponds to the density $|\psi(x,t)|^2$ of the ground truth solution, our approach (DSM), PINN, and t-VMC. The right panel presents statistics, including the particle's mean position and variance.

In addition, we consider an oscillator model with three non-interacting particles, which can be seen as a 3d system. The results are given in Table 2 and Appendix D.2.

5.2. Naive Sampling

To further evaluate our approach, we consider the following sampling scheme: it is possible to replace all measures in the expectations from Equation (14) with a Gaussian noise $\mathcal{N}(0,1)$. Minimizing this loss perfectly would imply that the PDE is satisfied for all values x, t. Table 2 shows worse quantitative results compared to our approach in the setting from Section 5.1. More detailed results, including the singular initial condition and 3d harmonic oscillator setting, are given in Appendix D.3.

5.3. Interacting System

Next, we consider a system of two interacting bosons in a harmonic trap with a soft contact term $V(x_1, x_2) = \frac{1}{2}m\omega^2(x_1^2 + x_2^2) + \frac{g}{2}\frac{1}{\sqrt{2\pi\sigma^2}}e^{-(x_1-x_2)^2/(2\sigma^2)}$ and initial condition $\psi_0 \propto e^{-m\omega^2 x^2/(2\hbar)}$. We use $\omega = 1$, $T = 1$, $\sigma^2 = 0.1$, and $N = 1000$. The term g controls interaction strength. When $g = 0$, there is no interaction, and ψ_0 is the ground state of the corresponding Hamiltonian $\mathcal{H}$. We use $g = 1$ in our simulations.

Figure 2 (c) shows simulation results: our method follows the corresponding ground truth while PINN fails over time. As t increases, the variance of X_i for PINN either decreases or remains relatively constant, contrasting with the dynamics that exhibit more divergent behavior. We hypothesize that such discrepancy in the performance of PINN, particularly in matching statistics, is due to the design choice. Specifically, the output predictions, $\psi(x_i, t)$, made by PINNs are not constrained to adhere to physical meaningfulness, meaning $\int_{\mathbb{R}^d} \left|\psi(x,t)\right|^2 \mathrm{d}x$ does not always equal 1, making uncontrolled statistics.

As for the t-VMC baseline, the results are a good qualitative approximation to the ground truth. The t-VMC ansatz representation comprises Hermite polynomials with two-body interaction terms (Carleo et al., 2017), scaling quadratically with the number of basis functions. This representation inherently incorporates knowledge about the ground truth solution. However, even when using the same number of samples and time steps as our DSM approach, t-VMC does not achieve the same level of accuracy, and the t-VMC approach does not perform well beyond $d = 3$ (see Appendix D.5). We anticipate the performance of t-VMC will further deteriorate for larger systems due to the absence of higher-order interactions in the chosen ansatz. We opted for this polynomial representation for scalability and because our experiments with neural network ansatzes (Schmitt & Heyl, 2020) did not yield satisfactory results for any d. Additional details are provided in Appendix C.2.

5.3.1. DSM in Higher Dimensions

To verify that our method can yield reasonable outputs for large many-body systems, we perform experiments on a 100 particle version of the interacting boson system. While ground truth is unavailable for a system of such a large scale, we perform a partial validation of our results by analyzing

Table 2. Results for different harmonic oscillator settings. In the 3d setting, the reported errors are averaged across all dimensions. The **best** results are in bold. [5]

SETTING	MODEL	$\mathcal{E}_m(X_i)\downarrow$	$\mathcal{E}_v(X_i)\downarrow$	$\mathcal{E}(v)\downarrow$	$\mathcal{E}(u)\downarrow$
$d=1$, $S_0(x)\equiv 0$	PINN	0.877 ± 0.263	0.766 ± 0.110	24.153 ± 3.082	4.432 ± 1.000
	DSM	$\mathbf{0.079 \pm 0.007}$	$\mathbf{0.019 \pm 0.005}$	$\mathbf{1.7 \times 10^{-4} \pm 4.9 \times 10^{-5}}$	$\mathbf{2.7 \times 10^{-5} \pm 4.9 \times 10^{-6}}$
	GAUSSIAN SAMPLING	0.355 ± 0.038	0.460 ± 0.039	8.478 ± 4.651	2.431 ± 0.792
$d=1$, $S_0(x)=-5x$	PINN	2.626 ± 0.250	0.626 ± 0.100	234.926 ± 57.666	65.526 ± 8.273
	DSM	$\mathbf{0.268 \pm 0.036}$	$\mathbf{0.013 \pm 0.008}$	$\mathbf{1.4 \times 10^{-5} \pm 5.5 \times 10^{-6}}$	$\mathbf{2.5 \times 10^{-5} \pm 3.8 \times 10^{-6}}$
	GAUSSIAN SAMPLING	0.886 ± 0.137	0.078 ± 0.013	73.588 ± 6.675	16.298 ± 6.311
$d=3$, $S_0(x)\equiv 0$	DSM (NELSONIAN)	$\mathbf{0.080 \pm 0.015}$	$\mathbf{0.016 \pm 0.007}$	$\mathbf{8.1 \times 10^{-5} \pm 2.8 \times 10^{-5}}$	$\mathbf{4.0 \times 10^{-5} \pm 2.2 \times 10^{-5}}$
	DSM (GRAD DIV)	$\mathbf{0.075 \pm 0.004}$	$\mathbf{0.015 \pm 0.004}$	$\mathbf{6.2 \times 10^{-5} \pm 2.2 \times 10^{-5}}$	$\mathbf{3.9 \times 10^{-5} \pm 2.9 \times 10^{-5}}$
	GAUSSIAN SAMPLING	0.423 ± 0.090	4.743 ± 0.337	6.505 ± 3.179	3.207 ± 0.911
$d=2$, INTERACTING SYSTEM	PINN	0.258 ± 0.079	1.937 ± 0.654	20.903 ± 7.676	10.210 ± 3.303
	DSM	$\mathbf{0.092 \pm 0.004}$	$\mathbf{0.055 \pm 0.015}$	$\mathbf{7.6 \times 10^{-5} \pm 1.0 \times 10^{-5}}$	$\mathbf{6.6 \times 10^{-5} \pm 2.8 \times 10^{-5}}$
	T-VMC	0.103 ± 0.007	0.109 ± 0.023	$2.9 \times 10^{-3} \pm 2.4 \times 10^{-4}$	$3.5 \times 10^{-4} \pm 0.8 \times 10^{-4}$

how the estimated densities change at $x = 0$ as a function of the interaction strength g. Scaling our method to many particles is straightforward, as we only need to adjust the neural network input size and possibly other parameters, such as a hidden dimension size. The obtained results in Figure 3 suggest that the time evolution is at least qualitatively reasonable since the one-particle density decays more quickly with increasing interaction strength g. In particular, this value should be higher for overlapping particles (a stable system with a low g value) and lower for moving apart particles (a system with a stronger interaction g). Furthermore, the low training loss of 10^{-2} order achieved by our model suggests that it is indeed representing a process consistent with Schrödinger equation, even for these large-scale systems. This experiment demonstrates our ability to scale the DSM approach to large interacting systems easily while providing partial validation of the results through the qualitative analysis of the one-particle density and its dependence on the interaction strength.

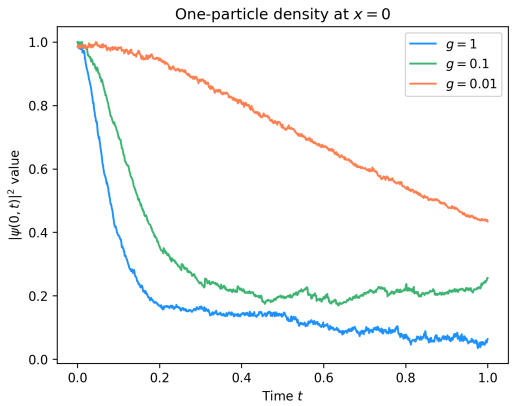


Figure 3. The one-particle density of a system of 100 interacting bosons for varying interaction strength g. For a weaker interaction, the one-particle density is higher, indicating a more stable particle configuration. Conversely, for a stronger interaction, this value decreases, suggesting a more dispersed particle behavior.

5.4. Computational and Memory Complexity

5.4.1. NON-INTERACTING SYSTEM

We measure training time per epoch and total train time for two versions of the DSM algorithm for $d = 1, 3, 5, 7, 9$: the Nelsonian one and our version. The experiments are conducted using the harmonic oscillator model with $S_0(x) \equiv 0$ from Section 5.1. The results are averaged across 30 runs. In this setting, the Hamiltonian is separable in the dimensions, and the problem has a linear scaling in d. However, given no prior knowledge about that, traditional numerical solvers and PINNs would suffer from exponential growth in data when tackling this task. Our method does not rely on a grid in x, and avoids computing the Laplacian in the loss function. That establishes lower bounds on the computational complexity of our method, and this bound is sharp for this particular problem. The advantageous behavior of our method is observed without any reliance on prior knowledge about the problem's nature.

Time per epoch The left panel of Figure 4 illustrates the scaling of time per iteration for both the Nelsonian formulation and our proposed approach. The time complexity exhibits a quadratic scaling trend for the Nelsonian version, while our method achieves a more favorable linear scaling behavior with respect to the problem dimension. These empirical observations substantiate our analytical complexity analysis.

Total training time The right panel of Figure 4 demonstrates the total training time of our version versus the prob-

[5] The difference between the mean errors of the DSM approach and other methods is statistically significant with a p-value < 0.001 measured by the one-sided Welsh t-test. Each model is trained and evaluated 10 times independently.

lem dimension. We train our models until the training loss reaches a threshold of 2.5×10^{-5}. We observe that the total training time exhibits a linear scaling trend as the dimensionality d increases. The performance errors are presented in Appendix D.4.

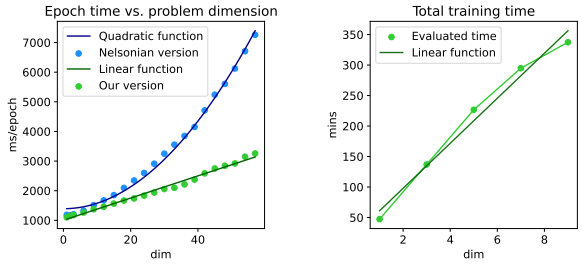


Figure 4. Empirical complexity evaluation of our method for the non-interacting system.

5.4.2. Interacting System

We study the scaling capabilities of our DSM approach in the setting from Section 5.3, comparing the performance of our algorithm with a numerical solver based on the Crank–Nicolson method. Table 4 shows training time, time per epoch, and memory usage for our method. Table 3 reports time and memory usage of the Crank–Nicolson method solver. More details and illustrations of obtained solutions are given in Appendix D.5.

Memory DSM memory usage and time per epoch grow linearly in d (according to our theory and evident in our numerical results) in contrast to the Crank-Nikolson solver, whose memory usage grows exponentially since discretization matrices are of $N^d \times N^d$ size. As a consequence, we are unable to execute the Crank-Nicolson method for $d > 4$ on our computational system due to memory constraints. The results show that our method is far more memory efficient for larger d.

Compute time While the total compute times of our DSM method, including training, are longer than those of the Crank-Nicolson solver for smaller values of d, the scaling trends suggest a computational advantage as d increases. In general, DSM is expected to scale quadratically with the problem dimension as there are pairwise interactions in our potential function.

Table 3. Time (s) to get a solution and memory usage (Gb) of the Crank-Nicolson method for different problem dimensions (interacting bosons).

	$d = 2$	$d = 3$	$d = 4$
Time	0.75	35.61	2363
Memory usage	7.4	10.6	214

Table 4. Training time (s), time per epoch (s/epoch), and memory usage (Gb) of our DSM method for different problem dimensions (interacting bosons).

	$d = 2$	$d = 3$	$d = 4$	$d = 5$
Training time	1770	3618	5850	9240
Time per epoch	0.52	1.09	1.16	1.24
Memory usage	17.0	22.5	28.0	33.5

6. Discussion and Limitations

This paper considers the simplest case of the linear spinless Schrödinger equation on a flat manifold $\mathbb{R}^d$ with a smooth potential. For many practical setups, such as quantum chemistry, quantum computing, or condensed matter physics, our approach should be modified, e.g., by adding a spin component or by considering some approximation and, therefore, requires additional validations that are beyond the scope of this work. We have shown evidence of adaptation of our method to one kind of low-dimensional structure, but this paper does not explore a broader range of systems with low latent dimension.

7. Conclusion

We develop a new algorithm for simulating quantum mechanics that addresses the curse of dimensionality by leveraging the latent low-dimensional structure of the system. This approach is based on a modification of the stochastic mechanics theory that establishes a correspondence between the Schrödinger equation and a diffusion process. We learn the drifts of this diffusion process using deep learning to sample from the corresponding quantum density. We believe that our approach has the potential to bring to quantum mechanics simulation the same progress that deep learning has enabled in artificial intelligence. We provide future work discussion in Appendix I.

Acknowledgements

The authors gratefully acknowledge the support of DOE DE-SC0022232, NSF DMS-2023109, NSF PHY2317138, NSF 2209892, and the University of Chicago Data Science Institute. Peter Y. Lu gratefully acknowledges the support of the Eric and Wendy Schmidt AI in Science Postdoctoral Fellowship, a Schmidt Futures program.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Alvarez, O. String theory and holomorphic line bundles. In *7th Workshop on Grand Unification: ICOBAN 86*, 9 1986.

Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

Austin, B. M., Zubarev, D. Y., and Lester Jr, W. A. Quantum Monte Carlo and related approaches. *Chemical reviews*, 112(1):263–288, 2012.

Aziznejad, S., Gupta, H., Campos, J., and Unser, M. Deep neural networks with trainable activations and controlled Lipschitz constant. *IEEE Transactions on Signal Processing*, 68:4688–4699, 2020.

Baldi, P. and Baldi, P. *Stochastic calculus*. Springer, 2017.

Barker, J. A. A quantum-statistical Monte Carlo method; path integrals with boundary conditions. *The Journal of Chemical Physics*, 70(6):2914–2918, 1979.

Bellman, R. E. *Dynamic programming*. Princeton university press, 2010.

Blanchard, P., Combe, P., Sirugue, M., and Sirugue-Collin, M. Stochastic jump processes associated with Dirac equation. In *Stochastic Processes in Classical and Quantum Systems: Proceedings of the 1st Ascona-Como International Conference, Held in Ascona, Ticino (Switzerland), June 24–29, 1985*, pp. 65–86. Springer, 2005.

Boffi, N. M. and Vanden-Eijnden, E. Probability flow solution of the Fokker-Planck equation, 2023.

Boghosian, B. M. and Taylor IV, W. Quantum lattice-gas model for the many-particle Schrödinger equation in d dimensions. *Physical Review E*, 57(1):54, 1998.

Bohm, D. A suggested interpretation of the quantum theory in terms of "hidden" variables. I. *Phys. Rev.*, 85:166–179, Jan 1952. doi: 10.1103/PhysRev.85.166. URL https://link.aps.org/doi/10.1103/PhysRev.85.166.

Bruna, J., Peherstorfer, B., and Vanden-Eijnden, E. Neural Galerkin scheme with active learning for high-dimensional evolution equations. *arXiv preprint arXiv:2203.01360*, 2022.

Buckdahn, R., Li, J., Peng, S., and Rainer, C. Mean-field stochastic differential equations and associated PDEs. *The Annals of Probability*, 45(2):824 – 878, 2017. doi: 10.1214/15-AOP1076. URL https://doi.org/10.1214/15-AOP1076.

Cances, E., Defranceschi, M., Kutzelnigg, W., Le Bris, C., and Maday, Y. Computational quantum chemistry: a primer. *Handbook of numerical analysis*, 10:3–270, 2003.

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

Carleo, G., Cevolani, L., Sanchez-Palencia, L., and Holzmann, M. Unitary dynamics of strongly interacting Bose gases with the time-dependent variational Monte Carlo method in continuous space. *Physical Review X*, 7(3): 031026, 2017.

Cliffe, K. A., Giles, M. B., Scheichl, R., and Teckentrup, A. L. Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients. *Computing and Visualization in Science*, 14:3–15, 2011.

Colin, S. and Struyve, W. Quantum non-equilibrium and relaxation to equilibrium for a class of de Broglie–Bohm-type theories. *New Journal of Physics*, 12(4):043008, 2010.

Corney, J. F. and Drummond, P. D. Gaussian quantum Monte Carlo methods for fermions and bosons. *Physical Review Letters*, 93(26), dec 2004. doi: 10.1103/physrevlett.93.260401. URL https://doi.org/10.1103%2Fphysrevlett.93.260401.

Dankel, T. G. Mechanics on manifolds and the incorporation of spin into Nelson's stochastic mechanics. *Archive for Rational Mechanics and Analysis*, 37:192–221, 1970.

De Angelis, G., Rinaldi, A., and Serva, M. Imaginary-time path integral for a relativistic spin-(1/2) particle in a magnetic field. *Europhysics Letters*, 14(2):95, 1991.

Del Moral, P. *Feynman-Kac formulae*. Springer, 2004.

Derakhshani, M. and Bacciagaluppi, G. On multi-time correlations in stochastic mechanics, 2022.

dos Reis, G., Engelhardt, S., and Smith, G. Simulation of McKean–Vlasov SDEs with super-linear growth. *IMA Journal of Numerical Analysis*, 42(1):874–922, 2022.

E, W. and Yu, B. The Deep Ritz method: A deep learning-based numerical algorithm for solving variational problems, 2017.

Eriksen, J. J. Mean-field density matrix decompositions. *The Journal of Chemical Physics*, 153(21):214109, 2020.

Fehrman, B., Gess, B., and Jentzen, A. Convergence rates for the stochastic gradient descent method for non-convex objective functions, 2019.

Ganesan, A., Coote, M. L., and Barakat, K. Molecular dynamics-driven drug discovery: leaping forward with confidence. *Drug discovery today*, 22(2):249–269, 2017.

Griewank, A. and Walther, A. *Evaluating Derivatives*. Society for Industrial and Applied Mathematics, second edition, 2008. doi: 10.1137/1.9780898717761. URL https://epubs.siam.org/doi/abs/10.1137/1.9780898717761.

Gronwall, T. H. Note on the derivatives with respect to a parameter of the solutions of a system of differential equations. *Annals of Mathematics*, 20(4):292–296, 1919. ISSN 0003486X. URL http://www.jstor.org/stable/1967124.

Grover, L. K. From Schrödinger's equation to the quantum search algorithm. *Pramana*, 56:333–348, 2001.

Guerra, F. Introduction to Nelson stochastic mechanics as a model for quantum mechanics. *The Foundations of Quantum Mechanics—Historical Analysis and Open Questions: Lecce, 1993*, pp. 339–355, 1995.

Gyöngy, I. Mimicking the one-dimensional marginal distributions of processes having an Itô differential. *Probability theory and related fields*, 71(4):501–516, 1986.

Han, J., Jentzen, A., and E, W. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34): 8505–8510, 2018.

Han, J., Zhang, L., and Weinan, E. Solving many-electron Schrödinger equation using deep neural networks. *Journal of Computational Physics*, 399:108929, 2019.

Heifetz, A. *Quantum mechanics in drug discovery*. Springer, 2020.

Hermann, J., Schätzle, Z., and Noé, F. Deep-neural-network solution of the electronic Schrödinger equation. *Nature Chemistry*, 12(10):891–897, 2020.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Hutchinson, M. F. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communication in Statistics- Simulation and Computation*, 18: 1059–1076, 01 1989. doi: 10.1080/03610919008812866.

Ilie, S., Jackson, K. R., and Enright, W. H. Adaptive time-stepping for the strong numerical solution of stochastic differential equations. *Numerical Algorithms*, 68(4):791–812, 2015.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Jiang, R. and Willett, R. Embed and Emulate: Learning to estimate parameters of dynamical systems with uncertainty quantification. *Advances in Neural Information Processing Systems*, 35:11918–11933, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kloeden, P. E. and Platen, E. *Stochastic differential equations*. Springer, 1992.

Lindgren, J. and Liukkonen, J. Quantum mechanics can be understood through stochastic optimization on spacetimes. *Scientific reports*, 9(1):19984, 2019.

Liu, R.-X., Tian, B., Liu, L.-C., Qin, B., and Lü, X. Bilinear forms, N-soliton solutions and soliton interactions for a fourth-order dispersive nonlinear Schrödinger equation in condensed-matter physics and biophysics. *Physica B: Condensed Matter*, 413:120–125, 2013.

Madala, V. C., Chandrasekaran, S., and Bunk, J. CNNs avoid curse of dimensionality by learning on patches. *IEEE Open Journal of Signal Processing*, 2023.

Manzhos, S. Machine learning for the solution of the Schrödinger equation. *Machine Learning: Science and Technology*, 1(1):013002, 2020.

May, J. P. *A concise course in algebraic topology*. University of Chicago press, 1999.

Muzellec, B., Sato, K., Massias, M., and Suzuki, T. Dimension-free convergence rates for gradient Langevin dynamics in RKHS, 2020.

Nakatsuji, H. Discovery of a general method of solving the Schrödinger and Dirac equations that opens a way to accurately predictive quantum chemistry. *Accounts of Chemical Research*, 45(9):1480–1490, 2012.

Neklyudov, K., Nys, J., Thiede, L., Carrasquilla, J., Liu, Q., Welling, M., and Makhzani, A. Wasserstein quantum Monte Carlo: a novel approach for solving the quantum many-body Schrödinger equation. *Advances in Neural Information Processing Systems*, 36, 2024.

Nelson, E. Derivation of the Schrödinger equation from Newtonian mechanics. *Phys. Rev.*, 150: 1079–1085, Oct 1966. doi: 10.1103/PhysRev.150.1079. URL https://link.aps.org/doi/10.1103/PhysRev.150.1079.

Nelson, E. The mystery of stochastic mechanics. *Unpublished manuscript*, 2005. URL https://web.math.princeton.edu/~nelson/papers/talk.pdf.

Nelson, E. *Dynamical theories of Brownian motion*, volume 106. Princeton university press, 2020.

Nüsken, N. and Richter, L. Interpolating between BSDEs and PINNs: deep learning for elliptic and parabolic boundary value problems. *arXiv preprint arXiv:2112.03749*, 2021a.

Nüsken, N. and Richter, L. Solving high-dimensional Hamilton–Jacobi–Bellman PDEs using neural networks: perspectives from the theory of controlled diffusions and measures on path space. *Partial differential equations and applications*, 2:1–48, 2021b.

Papageorgiou, A. and Traub, J. F. Measures of quantum computing speedup. *Physical Review A*, 88(2):022316, 2013.

Pfau, D., Spencer, J., de G. Matthews, A., and Foulkes, W. Ab-initio solution of the many-electron Schrödinger equation with deep neural networks. *Phys. Rev. Research*, 2:033429, 2020. doi: 10.1103/PhysRevResearch.2.033429. URL https://link.aps.org/doi/10.1103/PhysRevResearch.2.033429.

Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., and Liao, Q. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review. *International Journal of Automation and Computing*, 14(5):503–519, 2017.

Prieto, C. T. and Vitolo, R. On the geometry of the energy operator in quantum mechanics. *International Journal of Geometric Methods in Modern Physics*, 11(07):1460027, aug 2014. doi: 10.1142/s0219887814600275. URL https://doi.org/10.1142%2Fs0219887814600275.

Raginsky, M., Rakhlin, A., and Telgarsky, M. Non-convex learning via stochastic gradient Langevin dynamics: a nonasymptotic analysis, 2017.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Schlick, T. *Molecular modeling and simulation: an interdisciplinary guide*, volume 2. Springer, 2010.

Schmitt, M. and Heyl, M. Quantum many-body dynamics in two dimensions with artificial neural networks. *Physical Review Letters*, 125(10):100503, 2020.

Serkin, V. N. and Hasegawa, A. Novel soliton solutions of the nonlinear Schrödinger equation model. *Physical Review Letters*, 85(21):4502, 2000.

Serva, M. Relativistic stochastic processes associated to Klein-Gordon equation. *Annales de l'IHP Physique théorique*, 49(4):415–432, 1988.

Sinibaldi, A., Giuliani, C., Carleo, G., and Vicentini, F. Unbiasing time-dependent Variational Monte Carlo by projected quantum evolution. *arXiv preprint arXiv:2305.14294*, 2023.

Smith, G. D. and Smith, G. D. *Numerical solution of partial differential equations: finite difference methods*. Oxford university press, 1985.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Vicentini, F., Hofmann, D., Szabó, A., Wu, D., Roth, C., Giuliani, C., Pescia, G., Nys, J., Vargas-Calderón, V., Astrakhantsev, N., et al. NetKet 3: Machine learning toolbox for many-body quantum systems. *SciPost Physics Codebases*, pp. 007, 2022.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

Wallstrom, T. On the derivation of the Schrödinger equation from stochastic mechanics. *Foundations of Physics Letters*, 2:113–126, 03 1989. doi: 10.1007/BF00696108.

Wang, S., Yu, X., and Perdikaris, P. When and why PINNs fail to train: a neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022.

Warin, X. Nesting Monte Carlo for high-dimensional nonlinear PDEs. *Monte Carlo Methods and Applications*, 24 (4):225–247, 2018.

Weinan, E., Han, J., and Jentzen, A. Algorithms for solving high dimensional PDEs: from nonlinear Monte Carlo to machine learning. *Nonlinearity*, 35(1):278, 2021.

Woolley, R. and Sutcliffe, B. Molecular structure and the Born—Oppenheimer approximation. *Chemical Physics Letters*, 45(2):393–398, 1977.

Yan, J.-A. From Feynman-Kac formula to Feynman integrals via analytic continuation. *Stochastic processes and their applications*, 54(2):215–232, 1994.

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., Shao, Y., Zhang, W., Cui, B., and Yang, M.-H. Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796*, 2022.

Yao, Y.-X., Gomes, N., Zhang, F., Wang, C.-Z., Ho, K.-M., Iadecola, T., and Orth, P. P. Adaptive variational quantum dynamics simulations. *PRX Quantum*, 2(3): 030307, 2021.

A. Notation

- $\langle a, b\rangle = \sum_{i=1}^{d} a_i b_i$ for $a, b \in \mathbb{R}^d$ – a scalar product.
- $\|a\| = \sqrt{\langle a, a\rangle}$ for $a \in \mathbb{R}^d$ – a norm.
- $\mathrm{Tr}(A) = \sum_{i=1}^{d} a_{ii}$ for a matrix $A = \left[a_{ij}\right]_{i=1,j=1}^{d,d}$.
- $A(t), B(t), C(t), \ldots$ – stochastic processes indexed by time $t \geq 0$.
- $A_i, B_i, C_i, \ldots$ – approximations to those processes at a discrete time step i, $i = 1, \ldots, N$, where N is the number of discritization time points.
- a, b, c – other variables.
- $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots$ – quantum observables, e.g., $\mathbf{X}(t)$ – result of quantum measurement of the coordinate of the particle at moment t.
- $\rho_A(x, t)$ – a density probability of a process $A(t)$ at time t.
- $\psi(x, t)$ – a wave function.
- $\psi_0 = \psi(x, 0)$ – an initial wave function.
- $\rho(x, t) = \left|\psi(x, t)\right|^2$ – a quantum density.
- $\rho_0(x) = \rho(x, 0)$ – an initial probability distribution.
- $\psi(x, t) = \sqrt{\rho(x, t)} e^{iS(x,t)}$, where $S(x, t)$ – a single-valued representative of the phase of the wave function.
- $\nabla = \left(\frac{\partial}{\partial x_1}\cdot, \ldots, \frac{\partial}{\partial x_d}\cdot\right)$ – the gradient operator. If $f : \mathbb{R}^d \to \mathbb{R}^m$, then $\nabla f(x) \in \mathbb{R}^{d\times m}$ is the Jacobian of f, in the case of $m = 1$ we call it a gradient of f.
- $\nabla^2 = \left[\frac{\partial^2}{\partial x_i \partial x_j}\right]_{i=1,j=1}^{d,d}$ – the Hessian operator.
- $\nabla^2 \cdot A = \left[\frac{\partial^2}{\partial x_i \partial x_j} a_{ij}\right]_{i=1,j=1}^{d,d}$ for $A = \left[a_{ij}(x)\right]_{i=1,j=1}^{d,d}$.
- $\langle \nabla, \cdot\rangle$ – the divergence operator, e.g., for $f : \mathbb{R}^d \to \mathbb{R}^d$, we have $\langle \nabla, f(x)\rangle = \sum_{i=1}^{d} \frac{\partial}{\partial x_i} f_i(x)$.
- $\Delta = \mathrm{Tr}(\nabla^2)$ – the Laplace operator.
- m – a mass tensor (or a scalar mass).
- $\hbar$ – the reduced Planck's constant.
- $\partial_y = \frac{\partial}{\partial y}$ – a short-hand notation for a partial derivative operator.
- $\left[A, B\right] = AB - BA$ – a commutator of two operators. If one of the arguments is a scalar function, we consider a scalar function as a point-wise multiplication operator.
- $|z| = \sqrt{x^2 + y^2}$ for a complex number $z = x + iy \in \mathcal{C}, x, y \in \mathbb{R}$.
- $\mathcal{N}(\mu, C)$ – a Gaussian distribution with mean $\mu \in \mathbb{R}^d$ and covariance $C \in \mathbb{R}^{d\times d}$.
- $A \sim \rho$ means that A is a random variable with distribution ρ. We do not differentiate between "sample from" and "distributed as", but it is evident from context when we consider samples from distribution versus when we say that something has such distribution.
- δ_x – delta-distribution concentrated at x. It is a generalized function corresponding to the "density" of a distribution with a singular support $\{x\}$.

B. DSM Algorithm

We present detailed algorithmic descriptions of our method: Algorithm 2 for batch generation and Algorithm 3 for model training. During inference, distributions of X_i converge to $\rho = |\psi|^2$, thereby yielding the desired outcome. Furthermore, by solving Equation (7a) on points generated by the current best approximations of u, v, the method exhibits self-adaptation behavior. Specifically, it obtains its current belief where $X(t)$ is concentrated, updates its belief, and iterates accordingly. With each iteration, the method progressively focuses on the high-density regions of ρ, effectively exploiting the low-dimensional structure of the underlying solution.

Algorithm 2 GenerateBatch($u, v, \rho_0, \nu, T, B, N$) – sample trajectories

Physical hyperparams: T – time horizon, ψ_0 – initial wave-function.
Hyperparams: $\nu \geq 0$ – diffusion constant, $B \geq 1$ – batch size, $N \geq 1$ – time grid size.
$t_i = iT/N$ for $0 \leq i \leq N$
sample $X_{0j} \sim \left|\psi_0\right|^2$ for $1 \leq jB$
for $1 \leq i \leq N$ **do**
  sample $\xi_j \sim \mathcal{N}(0, I_d)$ for $1 \leq j \leq B$
  $X_{ij} = X_{(i-1)j} + \frac{T}{N}\left(v_\theta(X_{(i-1)j}, t_{i-1}) + \nu u_\theta(X_{(i-1)j}, t_{i-1})\right) + \sqrt{\frac{\nu\hbar T}{mN}}\xi_j$ for $1 \leq j \leq B$
end for
output $\left\{\left\{X_{ij}\right\}_{j=1}^{B}\right\}_{i=0}^{N}$

Algorithm 3 A training algorithm

Physical hyperparams: m – mass, $\hbar$ – reduced Planck constant, T – a time horizon, $\psi_0 : \mathbb{R}^d \to \mathbb{C}$ – an initial wave function, $V : \mathbb{R}^d \times [0, T] \to \mathbb{R}$ – potential.
Hyperparams: $\eta > 0$ – learning rate for backprop, $\nu > 0$ – diffusion constant, $B \geq 1$ – batch size, $M \geq 1$ – number of optimization steps, $N \geq 1$ – time grid size, $w_u, w_v, w_0 > 0$ – weights of losses.

Instructions:
$t_i = iT/N$ for $0 \leq i \leq N$
for $1 \leq \tau \leq M$ **do**
  $X = \text{GenerateBatch}(u_{\theta_{\tau-1}}, v_{\theta_{\tau-1}}, \psi_0, \nu, T, B, N)$
  define $L^u_\tau(\theta) = \frac{1}{(N+1)B}\sum_{i=0}^{N}\sum_{j=1}^{B}\left\|\partial_t u_\theta(X_{ij}, t_i) - \mathcal{D}_u[u_\theta, v_\theta, X_{ij}, t_i]\right\|^2$
  define $L^v_\tau(\theta) = \frac{1}{(N+1)B}\sum_{i=0}^{N}\sum_{j=1}^{B}\left\|\partial_t v_\theta(X_{ij}, t_i) - \mathcal{D}_v[u_\theta, v_\theta, X_{ij}, t_i]\right\|^2$
  define $L^0_\tau(\theta) = \frac{1}{B}\sum_{j=1}^{B}\left(\left\|u_\theta(X_{0j}, t_0) - u_0(X_{0j})\right\|^2 + \left\|v_\theta(X_{0j}, t_0) - v_0(X_{0j}, t_0)\right\|^2\right)$
  define $\mathcal{L}_\tau(\theta) = w_u L^u_\tau(\theta) + w_v L^v_\tau(\theta) + w_0 L^0_\tau(\theta)$
  $\theta_\tau = \text{OptimizationStep}(\theta_{\tau-1}, \nabla_\theta \mathcal{L}_\tau(\theta_{\tau-1}), \eta)$
end for
output $u_{\theta_M}, v_{\theta_M}$

C. Experiment Setup Details

C.1. Non-Interacting System

In our experiments, we set $m = 1$, $\hbar = 10^{-2}$[6], $\sigma^2 = 10^{-1}$. For the harmonic oscillator model, $N = 1000$ and the batch size $B = 100$; for the singular initial condition problem, $N = 100$ and $B = 100$. For evaluation, our method samples 10000 points per time step, and the observables are estimated from these samples; we run the model this way ten times.

C.1.1. A Numerical Solution

1d harmonic oscillator with $S_0(x) \equiv 0$: To evaluate our method's performance, we use a numerical solver that integrates the corresponding differential equation given the initial condition. We use SciPy library (Virtanen et al., 2020). The solution domain is $x \in [-2, 2]$ and $t \in [0, 1]$, where x is split into 566 points and t into 1001 time steps. This solution can be repeated d times for the d-dimensional harmonic oscillator problem.

[6]The value of the reduced Plank constant depends on the metric system that we use and, thus, for our evaluations we are free to choose any value.

1d harmonic oscillator with $S_0(x) = -5x$: We use the same numerical solver as for the $S_0(x) \equiv 0$ case. The solution domain is $x \in [-2, 2]$ and $t \in [0, 1]$, where x is split into 2829 points and t is split into 1001 time steps.

C.1.2. Architecture and Training Details

A basic NN architecture for our approach and the PINN is a feed-forward NN with one hidden layer with $\tanh$ activation functions. We represent the velocities u and v using this NN architecture with 200 neurons in the case of the singular initial condition. The training process takes about 7 mins. For $d = 1$, a harmonic oscillator with zero initial phase problem, there are 200 neurons for our method and 400 for the PINN; for $d = 3$ and more dimensions, we use 400 neurons. This rule holds for the experiments measuring total training time in Section 5.4. In a 1d harmonic oscillator with a non-zero initial phase problem, we use 300 hidden neurons in our models. In the experiments devoted to measuring time per epoch (from Section 5.4), the number of hidden neurons is fixed to 200 for all dimensions. We use the Adam optimizer (Kingma & Ba, 2014) with a learning rate 10^{-4}. In our experiments, we set $w_u = 1, w_v = 1, w_0 = 1$. For PINN evaluation, the test sets are the same as the grid for the numerical solver. In our experiments, we usually use a single NVIDIA A40 GPU. For the results reported in Section 5.4, we use an NVIDIA A100 GPU.

C.1.3. On Optimization

We use Adam optimizer (Kingma & Ba, 2014) in our experiments. Since the operators in Equation (8) are not linear, we may not be able to claim convergence to the global optima of such methods as SGD or Adam in the Neural Tangent Kernel (NTK) (Jacot et al., 2018) limit. Such proof exists for PINNs in Wang et al. (2022) due to the linearity of the Schrödinger equation (1). It is possible that non-linearity in the loss from Equation (14) requires non-convex methods to achieve theoretical guarantees on convergence to the global optima (Raginsky et al., 2017; Muzellec et al., 2020). Further research into NTK and non-linear PDEs is needed (Wang et al., 2022).

The only noise source in our loss Equation (14) comes from trajectory sampling. This fact contrasts sharply with generative diffusion models relying on score matching (Yang et al., 2022). In these models, the loss has $\mathcal{O}(\epsilon^{-1})$ variance as it implicitly attempts to numerically estimate the stochastic differential $\frac{X(t+\epsilon)-X(t)}{\epsilon}$ which leads to $\frac{1}{\sqrt{\epsilon}}$ contribution from increments of the Wiener process. In our loss, the stochastic differentials are evaluated analytically in Equation (8) avoiding such contributions; for details, see Nelson (1966; 2005). This leads to $\mathcal{O}(1)$ variance of the gradient and, thus, allows us to achieve fast convergence with smaller batches.

C.2. Interacting System

In our experiments, we set $m = 1$, $\hbar = 10^{-1}$, $\sigma^2 = 10^{-1}$. The number of time steps is $N = 1000$, and the batch size $B = 100$.

Numerical solution As a numerical solver, we use the QMSOLVE library [7]. The solution domain is $x \in [-1.5, 1.5]$ and $t \in [0, 1]$, where x is split into 100 points and t into 1001 time steps.

C.2.1. Architecture and Training Details

Instead of a multi-layer perceptron, we follow the design choice of Jiang & Willett (2022) to use residual connection blocks. In our experiments, we used the $\tanh$ as the activation function, set the hidden dimension to 300, and used the same architecture for both DSM and PINN. Empirically, we find out that this design choice leads to faster convergence in terms of training time. The PINN inputs are $N_0 = 10000$, $N_b = 1000$ data points, and $N_f = 1000000$ collocation points. We use Adam optimizer (Kingma & Ba, 2014) with a learning rate 10^{-4} in our experiments. We use loss weights $w_u = 1, w_v = 1, w_0 = 1$.

Permutation invariance Since our system comprises two identical bosons, we enforce symmetry for both the DSM and PINN models. Specifically, we sort the neural network inputs x to ensure the permutation invariance of the models. While this approach guarantees adherence to the physical symmetry property, it comes with a computational overhead from the sorting operation. For higher dimensional systems, avoiding such sorting may be preferable to reduce computational costs. However, for the two interacting particle system considered here, the performance difference between regular and

[7] https://github.com/quantum-visualizations/qmsolve

permutation-invariant architectures is not significant.

t-VMC ansatz To enable a fair comparison between our DSM approach and t-VMC, we initialize the t-VMC trial wave function with a complex-valued multi-layer perceptron architecture identical to the one employed in our DSM method. However, even after increasing the number of samples and reducing the time step, the t-VMC method exhibits poor performance with this neural network ansatz. This result suggests that, unlike our diffusion-based DSM approach, t-VMC struggles to achieve accurate results when utilizing simple off-the-shelf neural network architectures as the ansatz representation.

As an alternative ansatz, we employ a harmonic oscillator basis expansion, expressing the wave function as a linear combination of products of basis functions. This representation scales quadratically with the number of basis functions but forms a complete basis set for the two-particle problem. Using the same number of samples and time steps, this basis expansion approach achieves significantly better performance than our initial t-VMC baseline. However, it still does not match the accuracy levels attained by our proposed DSM method. This approach does not scale well naively to larger systems but can be adapted to form a 2-body Jastrow factor (Carleo et al., 2017). We expect this to perform worse for larger systems due to the lack of higher-order interactions in the ansatz. In our t-VMC experiments, we use the NETKET library (Vicentini et al., 2022) for many-body quantum systems simulation.

D. Experimental Results

D.1. Singular Initial Conditions

As a proof of concept, we consider a case of one particle $x \in \mathbb{R}^1$ with $V(x) \equiv 0$ and $\psi_0 = \delta_0$, $t \in [0, 1]$. Since δ-function is a generalized function, we must take a δ-sequence for training. The most straightforward approach is to take $\widetilde{\psi_0} = \frac{1}{(2\pi\alpha)^{\frac{1}{4}}} e^{-\frac{x^2}{4\alpha}}$ with $\alpha \to 0_+$. In our experiments we take $\alpha = \frac{\hbar^2}{m^2}$, yielding $v_0(x) \equiv 0$ and $u_0(x) = -\frac{\hbar x}{2m\alpha}$. Since ψ_0 is singular, we must set $\nu = 1$ during sampling. The analytical solution is known as $\psi(x, t) = \frac{1}{(2\pi t)^{\frac{1}{4}}} e^{-\frac{x^2}{4t}}$. So, we expect the standard deviation of $X(t)$ to grow as $\sqrt{t}$, and the mean value of $X(t)$ to be zero.

We do not compare our approach with PINNs since it is a simple proof of concept, and the analytical solution is known. Figure 5 summarizes the results of our experiment. Specifically, the left panel of the figure shows the magnitude of the density obtained with our approach alongside the true density. The right panel of Figure 5 shows statistics of X_t, such as mean and variance, and the corresponding error bars. The resulting prediction errors are calculated against the truth data for this problem and are measured at 0.008 ± 0.007 in the L_2-norm for the averaged mean and 0.011 ± 0.007 in the relative L_2-norm for the averaged variance of X_t. Our approach can accurately capture the behavior of the Schrödinger equation in the singular initial condition case.

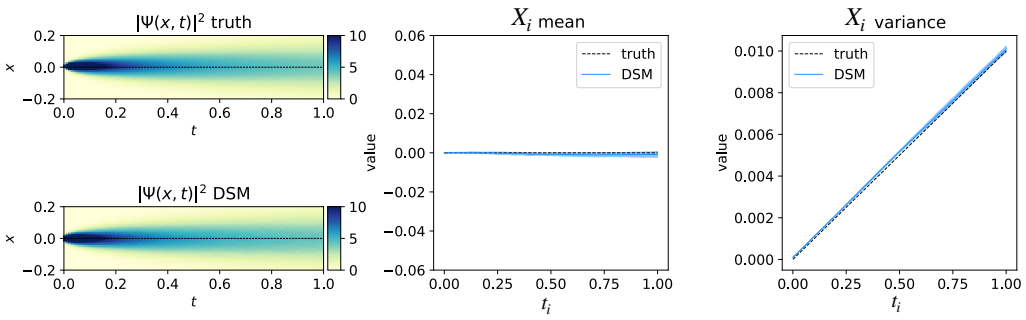


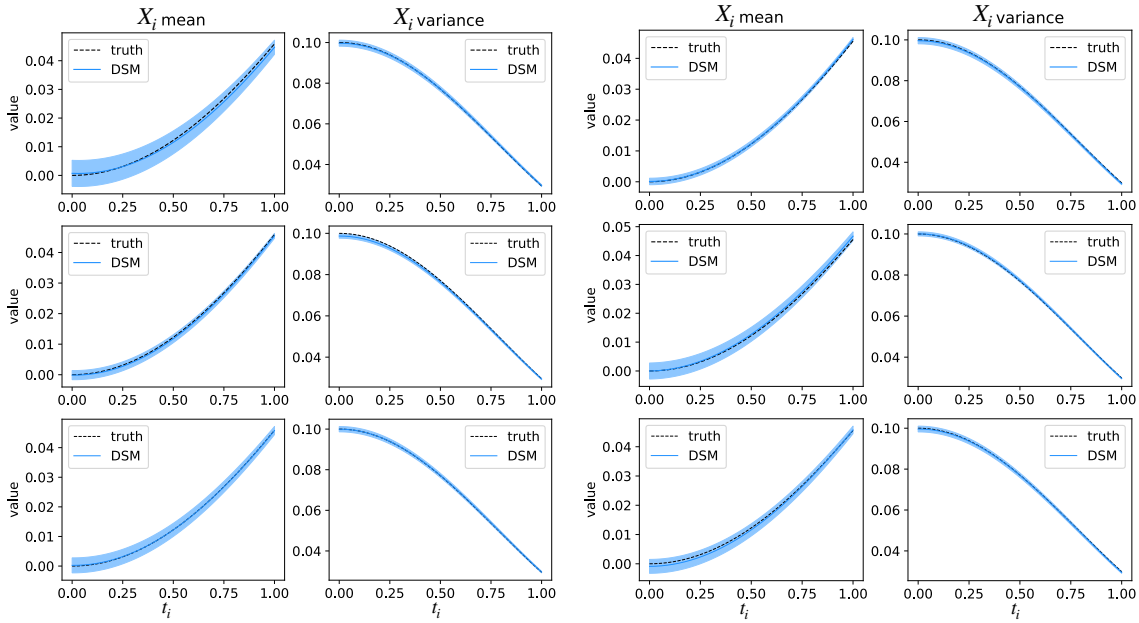
Figure 5. Results for the singular initial condition problem. DSM corresponds to our method.

D.2. 3D Harmonic Oscillator

We further explore our approach by considering the harmonic oscillator model with $S_0(x) \equiv 0$ with three non-interacting particles. This setting can be viewed as a 3d problem, where the solution is a 1d solution repeated three times. Due to computational resource limitations, we are unable to execute the PINN model. The number of collocation points should

grow exponentially with the problem dimension so that the PINN model converges. We have about 512 GB of memory but cannot store 60000^3 points. We conduct experiments comparing two versions of the proposed algorithm: the Nelsonian one and our version. Table 2 provides the quantitative results of these experiments. Our version demonstrates slightly better performance compared to the Nelsonian version, although the difference is not statistically significant. Empirically, our version requires more steps to converge compared to the Nelsonian version: 7000 vs. 9000 epochs correspondingly. However, the training time of the Nelsonian approach is about 20 mins longer than our approach's time.

Figure 6 demonstrates the obtained statistics with the proposed algorithm's two versions (Nelsonian and Gradient Divergence) for every dimension. Figure 7 compares the density function for every dimension for these two versions. Table 5 summarizes the error rates per dimension. The results suggest no significant difference in the performance of these two versions of our algorithm. The Gradient Divergence version tends to require more steps to converge, but it has quadratic time complexity in contrast to the cubic complexity of the Nelsonian version.



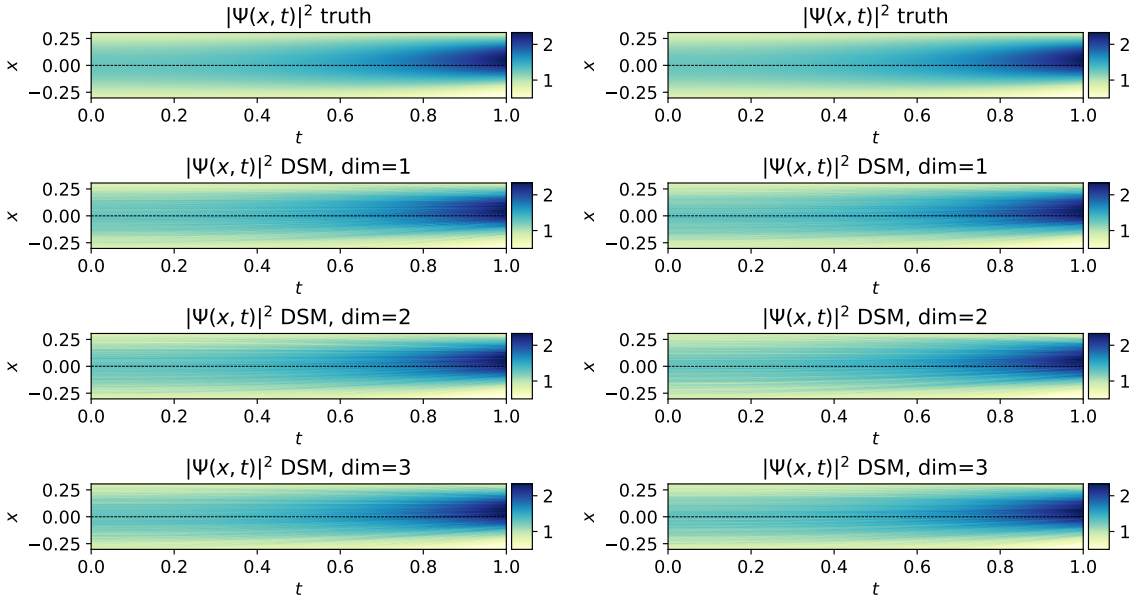
(a) The Nelsonian version.

(b) The Gradient Divergence version.

Figure 6. The obtained statistics for 3d harmonic oscillator using two versions of the proposed approach.

Table 5. The results for 3d harmonic oscillator with $S_0(x) \equiv 0$ using two versions of the proposed approach: the Nelsonian one uses the Laplacian operator in the training loss, the Gradient Divergence version is our modification that replaces Laplacian with gradient of divergence.

MODEL	$\mathcal{E}_m(X_i^{(1)}) \downarrow$	$\mathcal{E}_m(X_i^{(2)}) \downarrow$	$\mathcal{E}_m(X_i^{(3)}) \downarrow$	$\mathcal{E}_m(X_i) \downarrow$
DSM (NELSONIAN)	0.170 ± 0.081	0.056 ± 0.030	$\mathbf{0.073} \pm \mathbf{0.072}$	0.100 ± 0.061
DSM (GRADIENT DIVERGENCE)	$\mathbf{0.038} \pm \mathbf{0.023}$	$\mathbf{0.100} \pm \mathbf{0.060}$	0.082 ± 0.060	$\mathbf{0.073} \pm \mathbf{0.048}$
MODEL	$\mathcal{E}_v(X_i^{(1)}) \downarrow$	$\mathcal{E}_v(X_i^{(2)}) \downarrow$	$\mathcal{E}_v(X_i^{(3)}) \downarrow$	$\mathcal{E}_v(X_i) \downarrow$
DSM (NELSONIAN)	$\mathbf{0.012} \pm \mathbf{0.009}$	0.012 ± 0.009	0.011 ± 0.008	0.012 ± 0.009
DSM (GRADIENT DIVERGENCE)	$\mathbf{0.012} \pm \mathbf{0.010}$	$\mathbf{0.009} \pm \mathbf{0.005}$	$\mathbf{0.011} \pm \mathbf{0.010}$	$\mathbf{0.011} \pm \mathbf{0.008}$
MODEL	$\mathcal{E}(v^{(1)}) \downarrow$	$\mathcal{E}(v^{(2)}) \downarrow$	$\mathcal{E}(v^{(3)}) \downarrow$	$\mathcal{E}(v)) \downarrow$
DSM (NELSONIAN)	0.00013	0.00012	0.00012	0.00012
DSM (GRADIENT DIVERGENCE)	$\mathbf{4.346} \times \mathbf{10^{-5}}$	$\mathbf{4.401} \times \mathbf{10^{-5}}$	$\mathbf{4.700} \times \mathbf{10^{-5}}$	$\mathbf{4.482} \times \mathbf{10^{-5}}$
MODEL	$\mathcal{E}(u^{(1)}) \downarrow$	$\mathcal{E}(v^{(2)}) \downarrow$	$\mathcal{E}(v^{(3)}) \downarrow$	$\mathcal{E}(v) \downarrow$
DSM (NELSONIAN)	$\mathbf{4.441} \times \mathbf{10^{-5}}$	$\mathbf{2.721} \times \mathbf{10^{-5}}$	2.810×10^{-5}	$\mathbf{3.324} \times \mathbf{10^{-5}}$
DSM (GRADIENT DIVERGENCE)	6.648×10^{-5}	4.405×10^{-5}	$\mathbf{1.915} \times \mathbf{10^{-5}}$	4.333×10^{-5}



(a) The Nelsonian version.

(b) The Gradient Divergence version.

Figure 7. The density function for 3d harmonic oscillator using two versions of the proposed approach.

D.3. Naive Sampling

Figure 8 shows the performance of the Gaussian sampling approach applied to the harmonic oscillator and the singular initial condition setting. Table 6 compares results of all methods. Our approach converges to the ground truth while naive sampling does not.

Table 6. Error rates for different problem settings using two sampling schemes: our (DSM) and Gaussian sampling. Gaussian sampling replaces all measures in the expectations with Gaussian noise in Equation (14). The **best** result is in bold. These results demonstrate that our approach work better than the naïve sampling scheme.

PROBLEM	MODEL	$\mathcal{E}_m(X_i)\downarrow$	$\mathcal{E}_v(X_i)\downarrow$	$\mathcal{E}(v)\downarrow$	$\mathcal{E}(u)\downarrow$
SINGULAR IC	GAUSSIAN SAMPLING	0.043 ± 0.042	0.146 ± 0.013	1.262	0.035
	DSM	$\mathbf{0.008 \pm 0.007}$	$\mathbf{0.011 \pm 0.007}$	**0.524**	**0.008**
HARM OSC 1D, $S_0(x) \equiv 0$	GAUSSIAN SAMPLING	0.294 ± 0.152	0.488 ± 0.018	3.19762	1.18540
	DSM	$\mathbf{0.077 \pm 0.052}$	$\mathbf{0.011 \pm 0.006}$	**0.00011**	$\mathbf{2.811 \times 10^{-5}}$
HARM OSC 1D, $S_0(x) = -5x$	GAUSSIAN SAMPLING	0.836 ± 0.296	0.086 ± 0.007	77.57819	24.15156
	DSM	$\mathbf{0.223 \pm 0.207}$	$\mathbf{0.009 \pm 0.008}$	$\mathbf{1.645 \times 10^{-5}}$	$\mathbf{2.168 \times 10^{-5}}$
HARM OSC 3D, $S_0(x) \equiv 0$	GAUSSIAN SAMPLING	0.459 ± 0.126	5.101 ± 0.201	13.453	5.063
	DSM	$\mathbf{0.073 \pm 0.048}$	$\mathbf{0.011 \pm 0.008}$	$\mathbf{4.482 \times 10^{-5}}$	$\mathbf{4.333 \times 10^{-5}}$

D.4. Scaling Experiments for Non-Interacting System

We empirically estimate memory allocation on a GPU (NVIDIA A100) when training two versions of the proposed algorithm. In addition, we estimate the number of epochs until the training loss function is less than 10^{-2} for different problem dimensions. Figure 9(a) shows that the memory usage of the Gradient Divergence version grows linearly with the dimension while it grows quadratically in the Nelsonian version. We also empirically access the convergence speed of two versions of our approach. Figure 9(b) shows how many epochs are needed to make the training loss less than 1×10^{-2}. Usually, the Gradient Divergence version requires slightly more epochs to converge to this threshold than the Nelsonian one. The number of epochs is averaged across five runs. In both experiments, we consider the non-interacting harmonic oscillator setting from Section 5.4.1.

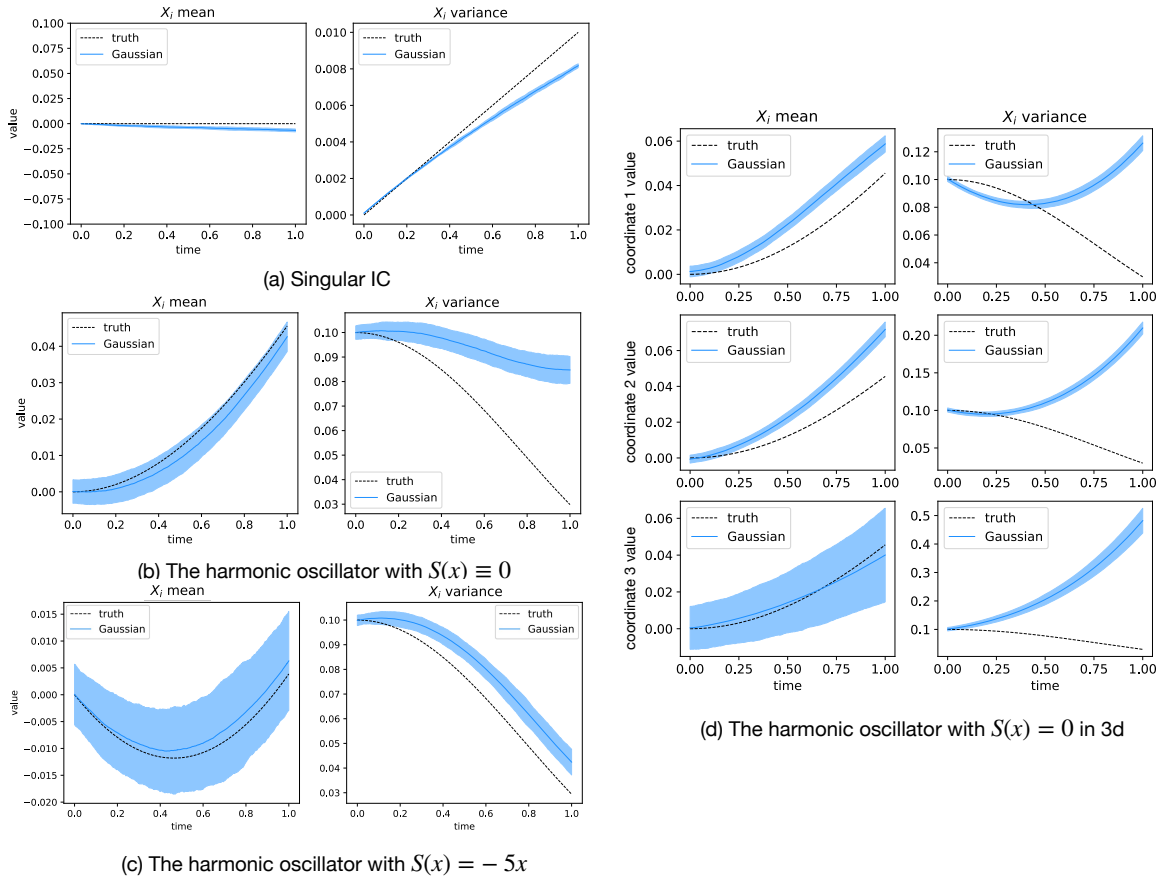


Figure 8. An illustration of produced trajectories using the naïve Gaussian sampling scheme as a comparison with the proposed approach. The obtained trajectories do not match the solution, while the results in our paper suggest that the proposed DSM approach converges better. Compare with Figures 2, 5, 6.

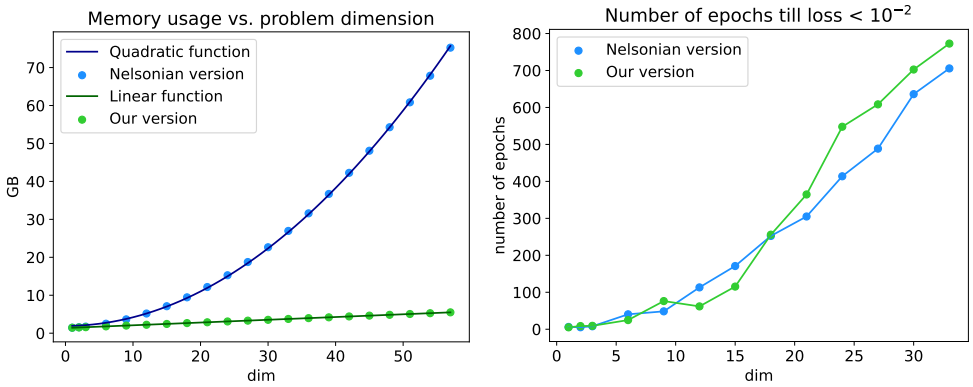
Also, we provide more details on the experiment measuring the total training time per dimensions $d = 1, 3, 5, 7, 9$. This experiment is described in Section 5.4.1, and the training time grows linearly with the problem dimension. Table 7 presents the error rates and train time. The results show that the proposed approach can perform well for every dimension while the train time scales linearly with the problem dimension.

Table 7. Training time and test errors for the harmonic oscillator model for different d.

d	$\mathcal{E}_m(X_i) \downarrow$	$\mathcal{E}_v(X_i) \downarrow$	$\mathcal{E}(v) \downarrow$	$\mathcal{E}(u) \downarrow$	TRAIN TIME
1	0.074 ± 0.052	0.009 ± 0.007	0.00012	2.809E-05	46M 20S
3	0.073 ± 0.048	0.010 ± 0.008	4.479×10^{-5}	3.946×10^{-5}	2H 18M
5	0.081 ± 0.057	0.009 ± 0.008	4.956×10^{-5}	4.000×10^{-5}	3H 10M
7	0.085 ± 0.060	0.011 ± 0.009	5.877×10^{-5}	4.971×10^{-5}	3H 40M
9	0.096 ± 0.081	0.011 ± 0.009	7.011×10^{-5}	6.123×10^{-5}	4H 46M

D.5. Scaling Experiments for the Interacting System

This section provides more details on experiments from Section 5.4.2, where we investigate the scaling of the DSM approach for the interacting bosons system. We compare the performance of our algorithm with a numerical solver based on the Crank–Nicolson method (we modified the QMSOLVE library to work for $d > 2$) and t-VMC method. Our method reveals favorable scaling capabilities in the problem dimension compared to the Crank–Nicolson method as shown in Table 3 and



(a) GPU memory usage. (b) Number of epochs until the training loss $< 10^{-2}$.

Figure 9. Empirical complexity evaluation of two versions of the proposed method: memory usage and the number of epochs until the loss is less than the threshold.

Table 4.

Figure 10 shows generated density functions for our DSM method and t-VMC approach. The proposed DSM approach demonstrates robust performance, accurately following the ground truth and providing reasonable predictions for $d = 3, 4, 5$ interacting bosons. In contrast, when utilizing the t-VMC in higher dimensions, we observe a deterioration in the quality of the results. This limitation is likely attributed to the inherent difficulty in accurately representing higher-order interactions with the ansatz employed in the t-VMC approach, as discussed in Section 5.3. Consequently, as the problem dimension grows, the lack of sufficient interaction terms in the ansatz and numerical instabilities in the solver become increasingly problematic, leading to artifacts in the density plots as time evolves. The relative error between the ground truth and predicted densities is 0.023 and 0.028 for the DSM and t-VMC approaches, respectively, in the 3d case. This trend persists in the 4d case, where the DSM's relative error is 0.073, compared to the t-VMC's higher relative error of 0.089 (when compared with a grid-based Crank-Nikolson solver with $N = 60$ grid points in each dimension). While we do not have the baseline for $d = 5$, we believe DSM predictions are still reasonable. Our findings indicate that the t-VMC method can perform reasonably for low-dimensional systems, but its performance degrades as the number of interacting particles increases. This highlights the need for a scalable and carefully designed ansatz representation capable of capturing the complex behavior of particles in high-dimensional quantum systems.

As for the DSM implementation details, we fix hyperparameters and only change d: for example, the neural network size is 500, and the batch size is 100. We train our method until the average training loss becomes lower than a particular threshold (0.007). These numbers are reported for a GPU A40. The Crank-Nikolson method is run on the CPU.

D.6. Sensitivity Analysis

We investigate the impact of hyperparameters on the performance of our method for two systems: the 1d harmonic oscillator with $S_0(x) \equiv 0$ and two interacting bosons. Specifically, we explore different learning rates ($10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}$) and hidden layer sizes (200, 300, 400, 500) for the neural network architectures detailed in Section C. All models are trained for an equal number of epochs across every hyper-parameter setting, and the results are presented in Figure 11. For the two interacting bosons system, increasing the hidden layer size leads to lower error, although the difference between 300 and 500 neurons is marginal. In contrast, for the 1d harmonic oscillator, larger hidden dimensions result in slightly worse performance (which might be a sign of overfitting for this simple problem), but the degradation is not substantial. As for the learning rate, a higher value consistently yields poorer performance for both systems. A large learning rate can cause the weight updates to overshoot the optimal values, leading to instability and failure to converge to a good solution. Nevertheless, all models achieve reasonable performance, even with the highest learning rate of 10^{-2}. Overall, according to the $\mathcal{E}_m(X_i)$ metric, our experiments demonstrate that our method exhibits robustness to varying hyper-parameter choices.

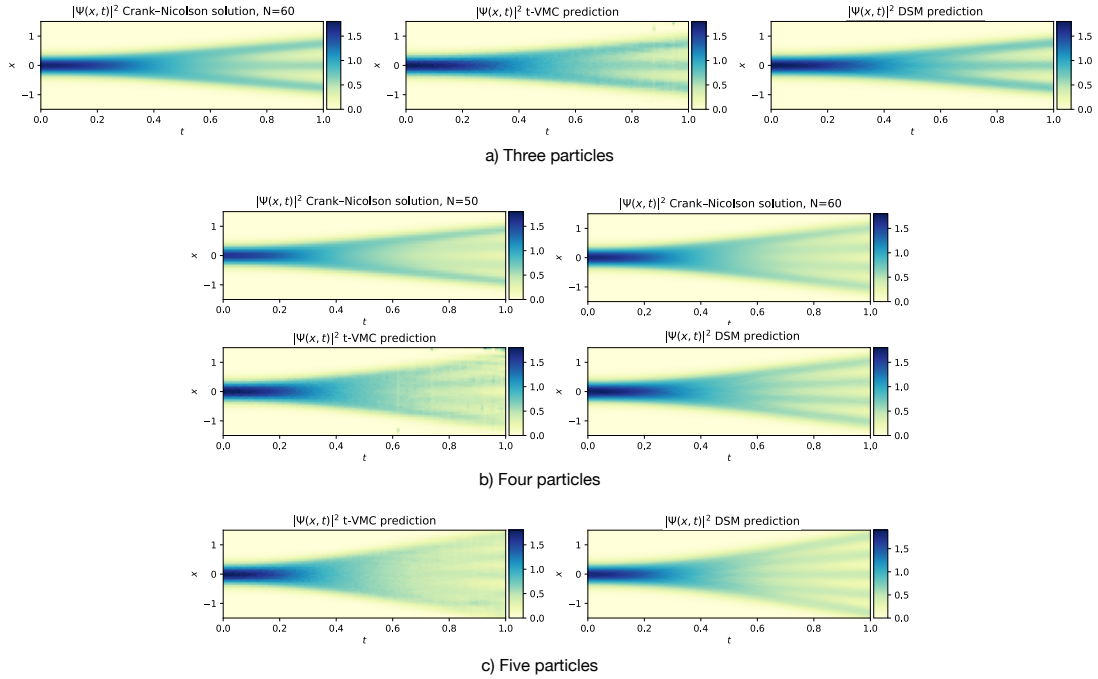


Figure 10. Probability density plots for different numbers of interacting particles d. For five particles, our computer system does not allow running the Crank-Nicolson solver.

E. Stochastic Mechanics

We show a derivation of the equations stochastic mechanics from the Schrödinger one. For full derivation and proof of equivalence, we refer the reader to the work of Nelson (1966).

E.1. Stochastic Mechanics Equations

Let's consider a polar decomposition of a wave function $\psi = \sqrt{\rho}e^{iS}$. Observe that for $\partial \in \{\partial_t, \partial_{x_i}\}$, we have

$$\partial\psi = (\partial\sqrt{\rho})e^{iS} + (i\partial S)\psi = \frac{\partial\rho}{2\sqrt{\rho}}e^{iS} + (i\partial S)\psi = \frac{1}{2}\frac{\partial\rho}{\rho}\sqrt{\rho}e^{iS} + (i\partial S)\psi = \big(\frac{1}{2}\partial\log\rho + i\partial S\big)\psi,$$
$$\partial^2\psi = \partial\Big(\big(\frac{1}{2}\partial\log\rho + i\partial S\big)\psi\Big) = \Big(\frac{1}{2}\partial^2\log\rho + i\partial^2 S + \big(\frac{1}{2}\partial\log\rho + i\partial S\big)^2\Big)\psi.$$

Substituting it into the Schrödinger equation, we obtain the following:

$$i\hbar\big(\frac{1}{2}\partial_t\log\rho + i\partial_t S\big)\psi = -\frac{\hbar^2}{2m}\Big(\frac{1}{2}\Delta\log\rho + i\Delta S + \big\|\frac{1}{2}\nabla\log\rho + i\nabla S\big\|^2\Big)\psi + V\psi. \tag{17}$$

Dividing by ψ[8], and separating real and imaginary parts, we obtain

$$-\hbar\partial_t S = -\frac{\hbar^2}{2m}\Big(\frac{1}{2}\Delta\log\rho + \frac{1}{4}\|\log\rho\|^2 - \|\nabla S\|^2\Big) + V, \tag{18}$$
$$\frac{\hbar}{2}\partial_t\log\rho = -\frac{\hbar^2}{2m}\big(\Delta S + \langle\log\rho, \nabla S\rangle\big). \tag{19}$$

[8]We assume $\psi \neq 0$. Even though it may seem a restriction, we will solve the equations only for $X(t)$, which satisfy $\mathbb{P}\big(\psi(X(t), t) = 0\big) = 0$. So, we are allowed to assume this without loss of generality. The same cannot be said if we considered the PINN over a grid to solve our equations.

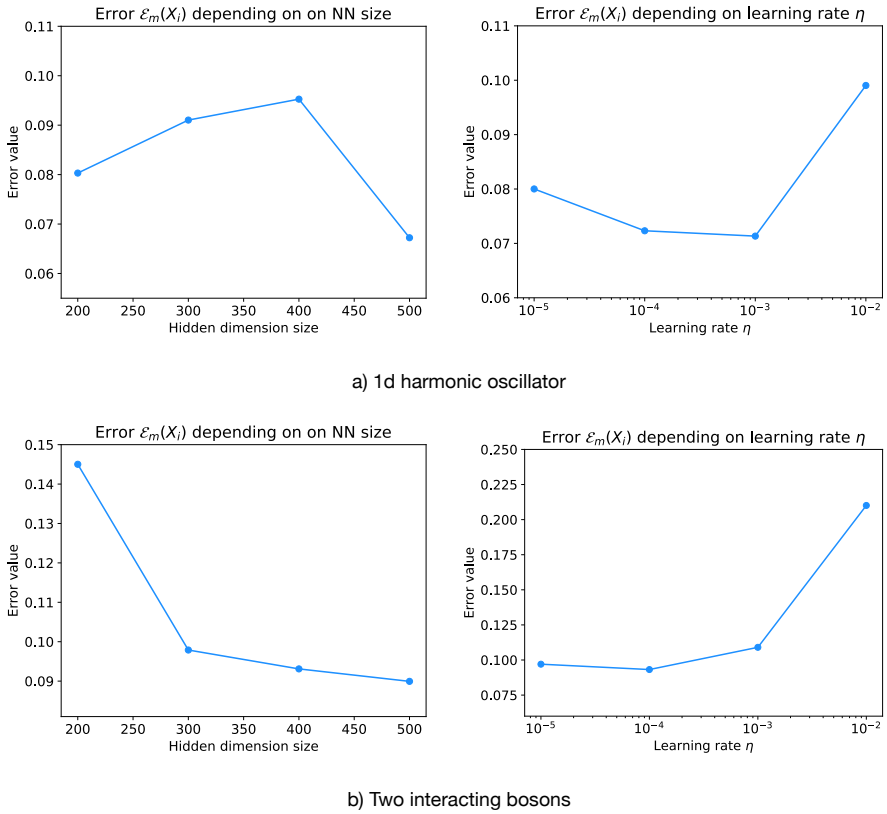


Figure 11. Sensitivity analysis of the neural network hyperparameters for the proposed method on two systems: (a) a 1D harmonic oscillator with $S_0(x) \equiv 0$, and (b) a system of two interacting bosons. The plots illustrate the impact of varying the hidden layer size and the learning rate on the model's performance, quantified by the $\mathcal{E}_m(X_i)$ error metric.

Noting that $\Delta = \langle \nabla, \nabla \cdot \rangle$ and substituting $v = \frac{\hbar}{m}\nabla S, u = \frac{\hbar}{2m}\log\rho$ to simplify, we obtain

$$m\frac{\hbar}{m}\partial_t S = \frac{\hbar}{2m}\langle \nabla, u\rangle + \frac{1}{2}\|u\|^2 - \frac{1}{2}\|v\|^2 - V, \tag{20}$$

$$\frac{\hbar}{2m}\partial_t \log\rho = -\frac{\hbar}{2m}\langle \nabla, v\rangle - \langle u, v\rangle. \tag{21}$$

Finally, by taking ∇ from both parts, noting that $\left[\nabla, \partial_t\right] = 0$ for scalar functions, and substituting u, v again, we arrive at

$$\partial_t v = -\frac{1}{m}\nabla V + \langle u, \nabla\rangle u - \langle v, \nabla\rangle v + \frac{\hbar}{2m}\nabla\langle \nabla, u\rangle, \tag{22}$$

$$\partial_t u = -\nabla\langle v, u\rangle - \frac{\hbar}{2m}\nabla\langle \nabla, v\rangle. \tag{23}$$

To get the initial conditions on the velocities of the process $v_0 = v(x, 0)$ and $u_0 = u(x, 0)$, we can refer to the equations that we used in the derivation

$$v(x, t) = \frac{\hbar}{m}\nabla S(x, t), \tag{24}$$

$$u(x, t) = \frac{\hbar}{2m}\nabla \log\rho(x, t) \tag{25}$$

So, we can get our initial conditions at $t = 0$ on $v_0(x) = \frac{\hbar}{m}\nabla S(x, 0), u_0(x) = \nu\nabla\log\rho_0(x)$, where $\rho_0(x) = \rho(x, 0)$.

For more detailed derivation and proof of equivalence of those two equations to the Schrödinger one, see Nelson (1966; 2005); Guerra (1995). Moreover, this equivalence holds for manifolds $\mathcal{M}$ with trivial second cohomology group as noted in Alvarez (1986); Wallstrom (1989); Prieto & Vitolo (2014).

E.2. Novel Equations of Stochastic Mechanics

We note that our equations differ from Guerra (1995); Nelson (1966). In Nelson (1966), we see

$$\partial_t v = -\frac{1}{m}\nabla V + \langle u, \nabla\rangle u - \langle v, \nabla\rangle v + \frac{\hbar}{2m}\Delta u, \tag{26a}$$

$$\partial_t u = -\nabla\langle v, u\rangle - \frac{\hbar}{2m}\nabla\langle\nabla, v\rangle; \tag{26b}$$

and in Guerra (1995), we see

$$\partial_t v = -\frac{1}{m}\nabla V + \langle u, \nabla\rangle u - \langle v, \nabla\rangle v + \frac{\hbar}{2m}\Delta u, \tag{27a}$$

$$\partial_t u = -\nabla\langle v, u\rangle - \frac{\hbar}{2m}\Delta v. \tag{27b}$$

Note that our equations (7a), (7b) do not directly use the second-order Laplacian operator Δ, as it appears for u in Equation (26a) and v in Equation (27b). The discrepancy between Nelson's and Guerra's equations seems to occur because the work by Nelson (2005) covers the case of the multi-valued S, and thus does not assume that $\left[\Delta, \nabla\right] = 0$ to transform $\nabla\langle\nabla, v\rangle = \nabla\langle\nabla, \nabla S\rangle$ into $\Delta(\nabla S)$ to make the equations work for the case of a non-trivial cohomology group of $\mathcal{M}$. However, Guerra (1995) does employ $\Delta(\nabla S)$ in their formulation. Naively computing the Laplacian Δ of u or v with autograd tools requires $\mathcal{O}(d^3)$ operations as it requires computing the full Hessian ∇^2. To reduce the computational complexity, we treat $\log\rho$ as a potentially multi-valued function, aiming to achieve a lower computational time of $\mathcal{O}(d^2)$ in the dimension d. Generally, we cannot swap Δ with $\nabla\langle\nabla, \cdot\rangle$ unless the solutions of the equation can be represented as full gradients of some function. This condition holds for stochastic mechanical equations but not for the Shrödinger one.

We derive equations different from both works and provide insights into why there are four different equivalent sets of equations (by changing Δ with $\nabla\langle\nabla, \cdot\rangle$ in both equations independently). From a numerical perspective, it is more beneficial to avoid Laplacian calculations. However, we notice that inference using equations from Nelson (1966) converges faster by iterations to the true u, v compared to our version. It comes at the cost of a severe slowdown in each iteration for $d \gg 1$, which diminishes the benefit since the overall training time to get comparable results decreases significantly.

E.3. Diffusion Processes of Stochastic Mechanics

Let's consider an arbitrary Ito diffusion process

$$\mathrm{d}X(t) = b(X(t), t)\mathrm{d}t + \sigma(X(t), t)\mathrm{d}\vec{W}, \tag{28}$$

$$X(0) \sim \rho_0, \tag{29}$$

where $W(t) \in \mathbb{R}^d$ is the standard Wiener process, $b : \mathbb{R}^d \times [0, T] \to \mathbb{R}^d$ is the drift function, and $\sigma : \mathbb{R}^d \times [0, T] \to \mathbb{R}^{d\times d}$ is a symmetric positive definite matrix-valued function called a diffusion coefficient. Essentially, $X(t)$ samples from $\rho_X = \mathrm{Law}(X(t))$ for each $t \in [0, T]$. Thus, we may wonder how to define b and σ to ensure $\rho_X = |\psi|^2$.

There is the forward Kolmogorov equation for the density ρ_X associated with this diffusion process:

$$\partial_t\rho_X = \langle\nabla, b\rho_X\rangle + \frac{1}{2}\mathrm{Tr}\big(\nabla^2 \cdot (\sigma\sigma^T\rho_X)\big). \tag{30}$$

Moreover, the diffusion process is time-reversible. This leads to the backward Kolmogorov equation:

$$\partial_t\rho_X = \langle\nabla, b^*\rho_X\rangle - \frac{1}{2}\mathrm{Tr}\big(\nabla^2 \cdot (\sigma\sigma^T\rho_X)\big), \tag{31}$$

where $b_i^* = b_i - \rho_X^{-1}\langle\nabla, \sigma\sigma^T e_i\rho_X\rangle$ with $e_{ij} = \delta_{ij}$ for $j \in \{1, \ldots, d\}$. Summing up those two equations, we obtain the following:

$$\partial_t\rho_X = \langle\nabla, v\rho_X\rangle, \tag{32}$$

where $v = \dfrac{b + b^*}{2}$ is so called probability current. This is the continuity equation for the Ito diffusion process from Equation (28). We refer to Anderson (1982) for details. We note that the same Equation (32) can be obtained with an arbitrary non-singular $\sigma(x, t)$ as long as $v = v(x, t)$ remains fixed.

Proposition E.1. *Consider arbitrary $\nu > 0$, denote $\rho = |\psi|^2$ and consider decomposition $\psi = \sqrt{\rho}e^{iS}$. Then the following process $X(t)$:*

$$\mathrm{d}X(t) = \big(\nabla S(X(t), t) + \frac{\nu\hbar}{2m}\nabla \log \rho(X(t), t)\big)\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\vec{W}, \tag{33}$$

$$X(0) \sim |\psi_0|^2, \tag{34}$$

satisfies $\mathrm{Law}(X(t)) = |\psi|^2$ *for any* $t > 0$.

Proof. We want to show that by choosing appropriately b, b_*, we can ensure that $\rho_X = |\psi|^2$. Let's consider the Schrödinger equation once again:

$$i\hbar\partial_t\psi = (-\frac{\hbar^2}{2m}\Delta + V)\psi, \tag{35}$$

$$\psi(\cdot, 0) = \psi_0 \tag{36}$$

where $\Delta = \mathrm{Tr}(\nabla^2) = \sum_{i=1}^d \frac{\partial^2}{\partial x_i^2}$ is the Laplace operator. The second cohomology is trivial in this case. So, we can assume that $\psi = \sqrt{\rho}e^{iS}$ with $S(x, t)$ is a single-valued function.

By defining the drift $v = \dfrac{\hbar}{m}\nabla S$, we can derive quantum mechanics continuity equation on density ρ:

$$\partial_t\rho = \langle\nabla, v\rho\rangle, \tag{37}$$

$$\rho(\cdot, 0) = \big|\psi_0\big|^2. \tag{38}$$

This immediately tells us what should be initial distribution ρ_0 and $\frac{b+b^*}{2}$ for the Ito diffusion process from Equation (28).

For now, the only missing parts for obtaining the diffusion process from the quantum mechanics continuity equation are to identify the term $\frac{b-b^*}{2}$ and the diffusion coefficient σ. Both of them should be related as $(b - b^*)_i = \rho^{-1}\langle\nabla, \sigma\sigma^T e_i\rho\rangle$. Thus, we can pick $\sigma \propto I_d$ to simplify the equations. Nevertheless, our results can be extended to any non-trivial diffusion coefficient. Therefore, by defining $u(x, t) = \dfrac{\hbar}{2m}\nabla \log \rho(x, t)$ and using arbitrary $\nu > 0$ we derive

$$\partial_t\rho = \langle\nabla, (v + \nu u)\rho\rangle + \frac{\nu\hbar}{2m}\Delta\rho. \tag{39}$$

Thus, we can sample from $\rho_X(x, t) \equiv \rho(x, t)$ using the diffusion process with $b(x, t) = v(x, t) + \nu u(x, t)$ and $\sigma(x, t) \equiv \frac{\nu\hbar}{m}I_d$:

$$\mathrm{d}X(t) = (v(X(t), t) + \nu u(X(t), t))\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\vec{W}, \tag{40}$$

$$X(0) \sim \big|\psi_0\big|^2. \tag{41}$$

□

To obtain numerical samples from the diffusion, one can use any numerical integrator, for example, the Euler-Maruyama integrator (Kloeden & Platen, 1992):

$$X_{i+1} = X_i + (v(X_i, t_i) + \nu u(X_i, t_i))\epsilon + \sqrt{\frac{\nu\hbar}{m}\epsilon}\mathcal{N}(0, I_d), \tag{42}$$

$$X_0 \sim \big|\psi_0\big|^2, \tag{43}$$

where $\epsilon > 0$ is a step size, $0 \leq i < \frac{T}{\epsilon}$. We consider this type of integrator in our work. However, integrators of higher order, e.g., Runge-Kutta family of integrators (Kloeden & Platen, 1992), can achieve the same integration error with larger $\epsilon > 0$; this approach is out of the scope of our work.

E.4. Interpolation between Bohmian and Nelsonian Pictures

We also differ from Nelson (1966) since we define u without ν. We bring it into the picture separately as a multiplicative factor:

$$\mathrm{d}X(t) = (v(X(t),t) + \nu u(X(t),t))\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\vec{W}, \tag{44}$$

$$X(0) \sim \left|\psi_0\right|^2 \tag{45}$$

This trick allows us to recover Nelson's diffusion when $\nu = 1$:

$$\mathrm{d}X(t) = (v(X(t),t) + u(X(t),t))\mathrm{d}t + \sqrt{\frac{\hbar}{m}}\mathrm{d}\vec{W}, \tag{46}$$

$$X(0) \sim \left|\psi_0\right|^2 \tag{47}$$

For cases of $|\psi_0|^2 > 0$ everywhere, e.g., if the initial conditions are Gaussian but not singular like δ_{x_0}, we can actually set $\nu = 0$ to obtain a deterministic flow:

$$\mathrm{d}X(t) = v(X(t),t)\mathrm{d}t, \tag{48}$$

$$X(0) \sim \left|\psi_0\right|^2. \tag{49}$$

This is the guiding equation in Bohr's pilot-wave theory (Bohm, 1952). The major drawback of using Bohr's interpretation is that ρ_X may not equal $\rho = |\psi|^2$, a phenomenon known as quantum non-equilibrium (Colin & Struyve, 2010). Though, under certain mild conditions (Boffi & Vanden-Eijnden, 2023) (one of which is $|\psi_0|^2 > 0$ everywhere) time marginals of such deterministic process $X(t)$ satisfy $\mathrm{Law}(X(t)) = \rho$ for each $t \in [0,T]$. As with the SDE case, it is unlikely that those trajectories are "true" trajectories. It only matters that their time marginals coincide with true quantum mechanical densities.

E.5. Computational Complexity

Proposition E.2 (Remark 4.1)**.** *The algorithmic complexity w.r.t. d of computing differential operators from Equations (8), (9) for velocities u, v is $\mathcal{O}(d^2)$.*

Proof. Computing a forward pass of u_θ, v_θ scales as $\mathcal{O}(d)$ by their design. What we need is to prove that Equations (8), (9) can be computed in $\mathcal{O}(d^2)$. We have two kinds of operators there: $\langle\nabla\cdot,\cdot\rangle$ and $\nabla\langle\nabla,\cdot\rangle$.

The first operator, $\langle\nabla\cdot,\cdot\rangle$, is a Jacobian-vector product. There exists an algorithm to estimate it with linear complexity, assuming the forward pass has linear complexity, as shown by Griewank & Walther (2008).

For the second operator, the gradient operator ∇ scales linearly with the problem dimension d. To estimate the divergence operator $\langle\nabla,\cdot\rangle$, we need to run automatic differentiation d times to obtain the full Jacobian and take its trace. This leads to a quadratic computational complexity of $\mathcal{O}(d^2)$ in the problem dimension. It is better than the naive computation of the Laplace operator Δ, which has a complexity of $\mathcal{O}(d^3)$ due to computing the full Hessian for each component of u_θ or v_θ. □

We assume that one of the dimensions when evaluating the d-dimensional functions involved in our method is parallelized by modern deep learning libraries. It means that empirically, we can see a linear $\mathcal{O}(d)$ scaling instead of the theoretical $\mathcal{O}(d^2)$ complexity.

F. On Strong Convergence

Let's consider a standard Wiener processes $\vec{W}^X, \vec{W}^Y$ in $\mathbb{R}^d$ and define $\vec{\mathcal{F}}_t$ as a filtration generated by $\left\{\left(\vec{W}^X(t'), \vec{W}^Y(t)\right) : t' \leq t\right\}$. Let $\overleftarrow{\mathcal{F}}_t$ be a filtration generated by all events $\left\{\left(\vec{W}^X(t'), \vec{W}^Y(t)\right) : t' \geq t\right\}$.

Assume that $u, v, \widetilde{u}, \widetilde{v} \in C^{2,1}(\mathbb{R}^d \times [0,T]; \mathbb{R}^d) \cap C_b^{1,0}(\mathbb{R}^d \times [0,T]; \mathbb{R}^d)$, where $C_b^{p,k}$ is a class of continuously differentiable functions with a uniformly bounded p-th derivative in a coordinate x and k-th continuously differentiable in t, $C^{p,k}$ analogously but without requiring bounded derivative. For $f : \mathbb{R}^d \times [0,T] \to \mathbb{R}^k$, we define $\|f\|_\infty = \operatorname{ess\,sup}_{t\in[0,T], x\in\mathbb{R}^d} \|f(x,t)\|$

and $\|\nabla f\|_\infty = \operatorname{ess\,sup}_{t\in[0,T],x\in\mathbb{R}^d}\|\nabla f(x,t)\|_{op}$, where $\|\cdot\|_{op}$ denotes an operator norm. Then we have the following equations:

$$\mathrm{d}X(t) = \big(\widetilde{v}(X(t),t) + \widetilde{u}(X(t),t)\big)\mathrm{d}t + \sqrt{\frac{\hbar}{m}}\mathrm{d}\overrightarrow{W}^X(t), \tag{50}$$

$$\mathrm{d}Y(t) = \big(v(Y(t),t) + u(Y(t),t)\big)\mathrm{d}t + \sqrt{\frac{\hbar}{m}}\mathrm{d}\overrightarrow{W}^Y(t), \tag{51}$$

$$X(0) \sim |\psi_0|^2, \tag{52}$$

$$Y(0) = X(0), \tag{53}$$

where u, v are true solutions to Equation (26). We have that $p_Y(\cdot,t) = \big|\psi(\cdot,t)\big|^2\ \forall t$ where p_Y is density of the process $Y(t)$. We have not specified yet a quadratic covariation of those two processes $\frac{\mathrm{d}\big[\overrightarrow{W}^X,\overrightarrow{W}^Y\big]_t}{\mathrm{d}t} = \lim_{\mathrm{d}t\to 0_+}\mathbb{E}\Big(\frac{\big(\overrightarrow{W}^X(t+\mathrm{d}t)-\overrightarrow{W}^X(t)\big)\big(\overrightarrow{W}^Y(t+\mathrm{d}t)-\overrightarrow{W}^Y(t)\big)}{\mathrm{d}t}\Big|\overrightarrow{\mathcal{F}}_t\Big)$. We specify it as $\mathrm{d}\big[\overrightarrow{W}^X,\overrightarrow{W}^Y\big]_t = I_d\mathrm{d}t$, and it allows to cancel some terms appearing in the equations. As for now, we will derive all results in the most general setting.

Let's define our loss functions:

$$L_1(\widetilde{v},\widetilde{u}) = \int_0^T \mathbb{E}^X\big\|\partial_t\widetilde{u}(X(t),t) - \mathcal{D}_u[\widetilde{v},\widetilde{u},x,t]\big\|^2\mathrm{d}t, \tag{54}$$

$$L_2(\widetilde{v},\widetilde{u}) = \int_0^T \mathbb{E}^X\big\|\partial_t\widetilde{v}(X(t),t) - \mathcal{D}_v[\widetilde{v},\widetilde{u},X(t),t]\big\|^2\mathrm{d}t, \tag{55}$$

$$L_3(\widetilde{u},\widetilde{v}) = \mathbb{E}^X\|\widetilde{u}(X(0),0) - u(X(0),0)\|^2, \tag{56}$$

$$L_4(\widetilde{u},\widetilde{v}) = \mathbb{E}^X\|\widetilde{v}(X(0),0) - v(X(0),0)\|^2. \tag{57}$$

Our goal is to show that for some constants $w_i > 0$, there is a natural bound $\sup_{0\le t\le T}\mathbb{E}\|X(t)-Y(t)\|^2 \le \sum w_i L_i(\widetilde{v},\widetilde{u})$.

F.1. Stochastic Processes

Consider a general Itô SDE defined using a drift process $F(t)$ and a covariance process $G(t)$, both predictable with respect to forward and backward flirtations $\overleftarrow{\mathcal{F}}_t$ and $\overrightarrow{\mathcal{F}}_t$:

$$\mathrm{d}Z(t) = F(t)\mathrm{d}t + G(t)\mathrm{d}\overrightarrow{W}, \qquad Z(0)\sim\rho_0. \tag{58}$$

Moreover, assume $\big[Z(t),Z(t)\big]_t = \mathbb{E}\int_0^t G^TG(t)\mathrm{d}t < \infty$, $\mathbb{E}\int_0^t \|F(t)\|^2\mathrm{d}t < \infty$. We denote by $\mathbb{P}_t^Z = \mathbb{P}(Z(t)\in\cdot)$ a law of the process $Z(t)$. Let's define a (extended) forward generate of the process as the linear operator satisfying

$$\overrightarrow{M}^f(t) = f(Z(t),t) - f(Z(0),0) - \int_0^t \overrightarrow{\mathcal{L}}^X f(Z(t),t) \text{ is } \overrightarrow{\mathcal{F}}_t\text{-martingale}. \tag{59}$$

Such an operator is uniquely defined and is called a forward generator associated with the process Z_t. Similarly, we define a (extended) backward generator $\overleftarrow{\mathcal{L}}^X$ as linear operator satisfying:

$$\overleftarrow{M}^f(t) = f(Z(t),t) - f(Z(0),0) - \int_0^t \overleftarrow{\mathcal{L}}^X f(Z(t),t) \text{ is } \overleftarrow{\mathcal{F}}_t\text{-martingale} \tag{60}$$

For more information on the properties of generators, we refer to Baldi & Baldi (2017).

Lemma F.1. *(Itô Lemma, (Baldi & Baldi, 2017, Theorem 8.1 and Remark 9.1))*

$$\overrightarrow{\mathcal{L}}^Z f(x,t) = \partial_t f(x,t) + \langle\nabla f(x,t),F(t)\rangle + \frac{\hbar}{2m}\mathrm{Tr}\big(G^T(t)\nabla^2 f(x,s)G(t)\big). \tag{61}$$

Lemma F.2. *Let $p_Z(x,t) = \frac{\mathrm{d}\mathbb{P}_t^Z}{\mathrm{d}x}$ be the density of the process with respect to standard Lebesgue measure on $\mathbb{R}^d$. Then*

$$\overleftarrow{\mathcal{L}}^Z f(x,t) = \partial_t f(x,t) + \langle \nabla f(x,t), F(t) - \frac{\hbar}{m}\nabla \log p_Z(x,t)\rangle - \frac{1}{2}\mathrm{Tr}\big(G^T(t)\nabla^2 f(x,s)G(t)\big). \tag{62}$$

Proof. We have the following operator identities:

$$\overleftarrow{\mathcal{L}}^Z = \big(\overrightarrow{\mathcal{L}}^Z\big)^* = p_Z^{-1}\big(\overrightarrow{\mathcal{L}}^X\big)^\dagger p_Z$$

where $\mathcal{A}^*$ is adjoint operator in $L_2(\mathbb{R}^d \times [0,T], \mathbb{P}^Z \otimes \mathrm{d}t)$ and $\mathcal{A}^\dagger$ is adjoint in $L_2(\mathbb{R}^d \times [0,T], \mathrm{d}x \otimes \mathrm{d}t)$. Using Itô lemma F.1 and grouping all terms yields the statement. □

Lemma F.3. *The following identity holds for any process $Z(t)$:*

$$\overrightarrow{\mathcal{L}}^Z \overleftarrow{\mathcal{L}}^Z x = \overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z x. \tag{63}$$

Proof. One needs to recognize that Equation (32) is the difference between two types of generators, we automatically have the following identity that holds for any process Z. □

Lemma F.4. *(Nelson Lemma, (Nelson, 2020))*

$$\mathbb{E}^Z\Big(f(Z(t),t)g(Z(t),t) - f(Z(0),t)g(Z(0),t)\Big) \tag{64}$$

$$= \mathbb{E}^Z \int_0^t \Big(\overrightarrow{\mathcal{L}}^Z f(Z(s),t)g(Z(s),t) + f(Z(s),t)\overleftarrow{\mathcal{L}}^Z g(Z(s),s)\Big)\mathrm{d}s \tag{65}$$

Lemma F.5. *It holds that:*

$$\mathbb{E}^Z\Big(\|Z(t)\|^2 - \|Z(0)\|^2\Big) \tag{66}$$

$$= \int_0^t \mathbb{E}^Z\Big(2\langle \overleftarrow{\mathcal{L}}^Z Z(0), Z(s)\rangle + 2\int_0^s \langle \overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z Z(z), Z(s)\rangle \mathrm{d}z\Big)\mathrm{d}s + \big[Z(t), Z(t)\big]_t \tag{67}$$

Proof. By using Itô Lemma F.1 for $f(x) = \|x\|^2$ and noting that $\overrightarrow{\mathcal{L}}^Z Z(t) = F(t)$ we immediately obtain:

$$\mathbb{E}^Z(\|Z(t)\|^2 - \|Z(0)\|^2) = \int_0^t \mathbb{E}\Big(2\langle \overrightarrow{\mathcal{L}}^Z Z(s), Z(s)\rangle + \mathrm{Tr}\big(G^T G(t)\big)\Big)\mathrm{d}s$$

Let's deal with the term $\int_0^t \langle \overrightarrow{\mathcal{L}}^Z Z(s), Z(s)\rangle \mathrm{d}s$. We have the following observation: $\overrightarrow{M}^F(z) = \overleftarrow{\mathcal{L}}^Z Z(s) - \overleftarrow{\mathcal{L}}^Z Z(0) - \int_0^s \overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z Z(z)\mathrm{d}z$ is $\overleftarrow{\mathcal{F}}_s$-martingale, thus

$$\int_0^t \langle \overrightarrow{\mathcal{L}}^Z Z(s), Z(s)\rangle \mathrm{d}s = \int_0^t \langle \overleftarrow{\mathcal{L}}^Z Z(0) + \int_0^s \big(\overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z Z(z) + \overleftarrow{M}^F(z)\big)\mathrm{d}z, Z(s)\rangle \mathrm{d}s,$$

The process $\overrightarrow{A}(s',s) = \int_{s'}^s \langle \overleftarrow{M}^F(z), Z(s)\rangle \mathrm{d}z$ is again $\mathcal{F}_{s'}^{\leftarrow}$-martingale for $s' \leq s$, which implies that $\mathbb{E}^Z \overrightarrow{A}(0,s) = 0$. Noting that $\mathbb{E}^Z \int_0^t \mathrm{Tr}\big(G^T(t)G(t)\big)\mathrm{d}t = \big[Z(t), Z(t)\big]_t$ yields the lemma. □

F.2. Adjoint Processes

Consider a process $X'(t)$ defined through time-reversed SDE:

$$\mathrm{d}X'(t) = \big(\widetilde{v}(X'(t),t) + \widetilde{u}(X'(t),t)\big)\mathrm{d}t + \sqrt{\frac{\hbar}{2m}}\mathrm{d}\overleftarrow{W}^X(t). \tag{68}$$

We call such process as adjoint to the process X. Lemma F.3 can be generalized to the pair of adjoint processes (X, X') in the following way and will be instrumental in proving our results.

Lemma F.6. *For any pair of processes $X(t), X'(t)$ such that the forward drift of X is of form $\widetilde{v}+\widetilde{u}$ and backward drift of X' is $\widetilde{v}-\widetilde{u}$:*

$$\overrightarrow{\mathcal{L}}^{X}\overleftarrow{\mathcal{L}}^{X'}x-\overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^{X}x=\overleftarrow{\mathcal{L}}^{X'}\overleftarrow{\mathcal{L}}^{X'}x-\overrightarrow{\mathcal{L}}^{X}\overrightarrow{\mathcal{L}}^{X}x. \tag{69}$$

with both sides being equal to 0 *if and only if X' is time reversal of X.*

Proof. Manual substitution of explicit forms of generators and drifts yields Equation (7b) for both cases. This equation is zero only if $\widetilde{u}=\frac{\hbar}{2m}\nabla\log p_X$ □

Lemma F.7. *The following bound holds:*

$$\Big\|\big(\overrightarrow{\mathcal{L}}^{X}+\overleftarrow{\mathcal{L}}^{X}\big)(\widetilde{u}-\frac{\hbar}{2m}\nabla\log p_X)\|\leq\Big\|\overrightarrow{\mathcal{L}}^{X}\overleftarrow{\mathcal{L}}^{X'}x-\overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^{X}x\Big\|+2\|\nabla\widetilde{v}\|_{\infty}\big\|\widetilde{u}-\frac{\hbar}{2m}\nabla\log p_X\big\|. \tag{70}$$

Proof. First, using Lemma F.6 we obtain:

$$\begin{aligned}
&\overrightarrow{\mathcal{L}}^{X}\overleftarrow{\mathcal{L}}^{X}x-\overleftarrow{\mathcal{L}}^{X}\overrightarrow{\mathcal{L}}^{X}x=0 && (71)\\
\iff\ &\overrightarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big)-\overleftarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)=0 && (72)\\
\iff\ &\overrightarrow{\mathcal{L}}^{X}\big((\widetilde{v}-\widetilde{u})+(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X)\big)-\overleftarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)=0 && (73)\\
\iff\ &\overrightarrow{\mathcal{L}}^{X}\big((\widetilde{v}-\widetilde{u})+(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X)\big)-\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}+\widetilde{u}\big)+\Big(\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}+\widetilde{u}\big)-\overleftarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)\Big)=0 && (74)\\
\iff\ &\overrightarrow{\mathcal{L}}^{X}\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big)+\overrightarrow{\mathcal{L}}^{X}(\widetilde{v}-\widetilde{u})-\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}+\widetilde{u}\big)+\Big(\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}+\widetilde{u}\big)-\overleftarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)\Big)=0. && (75)
\end{aligned}$$

Then, we note that:

$$\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}+\widetilde{u}\big)-\overleftarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)=\langle\frac{\hbar}{m}\nabla\log p_X-2\widetilde{u},\nabla(\widetilde{v}+\widetilde{u})\rangle. \tag{76}$$

This leads us to the following identity:

$$\begin{aligned}
&\overrightarrow{\mathcal{L}}^{X}\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big)+\overrightarrow{\mathcal{L}}^{X}(\widetilde{v}-\widetilde{u})-\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}+\widetilde{u}\big)+\langle\frac{\hbar}{m}\nabla\log p_X-2\widetilde{u},\nabla(\widetilde{v}+\widetilde{u})\rangle=0\\
\iff\ &\overrightarrow{\mathcal{L}}^{X}\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big)+\overrightarrow{\mathcal{L}}^{X}\overleftarrow{\mathcal{L}}^{X'}x-\overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^{X}x+\langle\frac{\hbar}{m}\nabla\log p_X-2\widetilde{u},\nabla(\widetilde{v}+\widetilde{u})\rangle=0.
\end{aligned}$$

Again by using Lemma F.6 to time-reversal X' we obtain:

$$\begin{aligned}
&\overleftarrow{\mathcal{L}}^{X}\overleftarrow{\mathcal{L}}^{X}x-\overrightarrow{\mathcal{L}}^{X}\overrightarrow{\mathcal{L}}^{X}x=0 && (77)\\
\iff\ &\overleftarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big)-\overrightarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)=0 && (78)\\
\iff\ &\overleftarrow{\mathcal{L}}^{X}\big((\widetilde{v}-\widetilde{u})+(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X)\big)-\overrightarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)=0 && (79)\\
\iff\ &\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}-\widetilde{u}\big)+\overleftarrow{\mathcal{L}}^{X}\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big)-\overrightarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)+\Big(\overleftarrow{\mathcal{L}}^{X}\big(\widetilde{v}-\widetilde{u}\big)-\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}-\widetilde{u}\big)\Big)=0 && (80)\\
\iff\ &\overleftarrow{\mathcal{L}}^{X}\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big)+\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}-\widetilde{u}\big)-\overrightarrow{\mathcal{L}}^{X}\big(\widetilde{v}+\widetilde{u}\big)-\langle\frac{\hbar}{m}\nabla\log p_X-2\widetilde{u},\nabla(\widetilde{v}-\widetilde{u})\rangle=0 && (81)\\
\iff\ &\overleftarrow{\mathcal{L}}^{X}\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big)+\overleftarrow{\mathcal{L}}^{X'}\overleftarrow{\mathcal{L}}^{X'}x-\overrightarrow{\mathcal{L}}^{X}\overrightarrow{\mathcal{L}}^{X}x-\langle\frac{\hbar}{m}\nabla\log p_X-2\widetilde{u},\nabla(\widetilde{v}-\widetilde{u})\rangle=0. && (82)
\end{aligned}$$

By using Lemma F.6 we thus derive:

$$\overleftarrow{\mathcal{L}}^X\big(2\widetilde{u} - \frac{\hbar}{m}\nabla \log p_X\big) + \overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}x - \overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^X x - \langle \frac{\hbar}{m}\nabla \log p_X - 2\widetilde{u}, \nabla(\widetilde{v} - \widetilde{u})\rangle = 0. \tag{83}$$

Summing up both identities, therefore, yields:

$$\Big(\overleftarrow{\mathcal{L}}^X + \overrightarrow{\mathcal{L}}^X\Big)\big(\widetilde{u} - \frac{\hbar}{2m}\nabla \log p_X\big) + \overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}x - \overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^X x + 2\langle \widetilde{u} - \frac{\hbar}{2m}\nabla \log p_X, \nabla\widetilde{v}\rangle = 0. \tag{84}$$

□

Theorem F.8. *The following bound holds:*

$$\sup_{0\leq t\leq T} \mathbb{E}^X\big\|\widetilde{u}(X(t),t) - \frac{\hbar}{2m}\nabla \log p_X(X(t),t)\big\|^2 \leq e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\big(L_3(\widetilde{v},\widetilde{u}) + L_2(\widetilde{v},\widetilde{u})\big). \tag{85}$$

Proof. We consider process $Z(t) = \widetilde{u}u(X(t),t) - \frac{\hbar}{2m}\nabla \log p_X(X(t),t)$. From Nelson's lemma F.4, we have the following identity:

$$\mathbb{E}^X\|\widetilde{u}(X(t),t) - \frac{\hbar}{2m}\nabla \log p_X(X(t),t)\|^2 - \mathbb{E}^X\|\widetilde{u}(X(0),0) - \frac{\hbar}{2m}\nabla \log p_X(X(0),0)\|^2 \tag{86}$$

$$= \mathbb{E}^X\int_0^t \langle u(X(s),s) - \frac{\hbar}{2m}\nabla \log p_X(X(s),s), \tag{87}$$

$$\big(\overrightarrow{\mathcal{L}}^X + \overleftarrow{\mathcal{L}}^X\big)\big(u(X(s),s) - \frac{\hbar}{2m}\nabla \log p_X(X(s),s)\big)\rangle \mathrm{d}s. \tag{88}$$

Note that $u \equiv \frac{\hbar}{2m}\nabla \log p_X(X(t),t)$. Thus, $\mathbb{E}^X\|\widetilde{u}(X(0),0) - \frac{\hbar}{2m}\nabla \log p_X(X(0),0)\|^2 = L_3(\widetilde{v},\widetilde{u})$. Using inequality $\langle a,b\rangle \leq \frac{1}{2}\big(\|a\|^2 + \|b\|^2\big)$ we obtain:

$$\mathbb{E}^X\|u(X(t),t) - \frac{\hbar}{2m}\nabla \log p_X(X(t),t)\|^2 - L_3(\widetilde{v},\widetilde{u}) \tag{89}$$

$$\leq \int_0^t \Big(\frac{1}{2}\mathbb{E}^X\|u(X(s),s) - \frac{\hbar}{2m}\nabla \log p_X(X(s),s)\|^2 \tag{90}$$

$$+ \frac{1}{2}\mathbb{E}^X\Big\|\big(\overrightarrow{\mathcal{L}}^X + \overleftarrow{\mathcal{L}}^X\big)\big(u(X(s),s) - \frac{\hbar}{2m}\nabla \log p_X(X(s),s)\big)\Big\|^2\Big)\mathrm{d}s \tag{91}$$

Using Lemma F.7, we obtain:

$$\mathbb{E}^X\|u(X(t),t) - \frac{\hbar}{2m}\nabla \log p_X(X(t),t)\|^2 - L_3(\widetilde{v},\widetilde{u}) \tag{92}$$

$$\leq \int_0^t \Big(\frac{1}{2}\mathbb{E}^X\|u(X(s),s) - \frac{\hbar}{2m}\nabla \log p_X(X(s),s)\|^2 \tag{93}$$

$$+ \Big\|\overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}x - \overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^X x\Big\|^2 + 4\|\nabla\widetilde{v}\|_\infty^2\big\|\widetilde{u} - \frac{\hbar}{2m}\nabla \log p_X\big\|^2\Big)\mathrm{d}s \tag{94}$$

Observe that $\int_0^t \mathbb{E}^X\Big\|\overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}x - \overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^X x\Big\|^2\mathrm{d}t \leq L_2(\widetilde{v},\widetilde{u})$, in fact, at $t = T$ it is equality as this is the definition of the loss L_2. Thus, we have:

$$\mathbb{E}^X\|u(X(t),t) - \frac{\hbar}{2m}\nabla \log p_X(X(t),t)\|^2 \tag{95}$$

$$\leq L_3(\widetilde{v},\widetilde{u}) + L_2(\widetilde{v},\widetilde{u}) + \int_0^t \big(\frac{1}{2} + 4\|\nabla\widetilde{v}\|_\infty\big)\mathbb{E}^X\|u(X(s),s) - \frac{\hbar}{2m}\nabla \log p_X(X(s),s)\|^2\mathrm{d}s. \tag{96}$$

Using integral Grönwall's inequality (Gronwall, 1919) yields the bound: $\mathbb{E}^X\|u(X(t),t) - \frac{\hbar}{2m}\nabla \log p_X(X(t),t)\|^2 \leq e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)t}\big(L_3(\widetilde{v},\widetilde{u}) + L_2(\widetilde{v},\widetilde{u})\big)$. □

F.3. Nelsonian Processes

Considering those two operators, we can rewrite the equations (26) alternatively as:

$$\frac{1}{2}\Big(\overrightarrow{\mathcal{L}}^Y\overleftarrow{\mathcal{L}}^Y x+\overleftarrow{\mathcal{L}}^Y\overrightarrow{\mathcal{L}}^Y x\Big)=-\frac{1}{m}\nabla V(x), \tag{97}$$

$$\frac{1}{2}\Big(\overrightarrow{\mathcal{L}}^Y\overleftarrow{\mathcal{L}}^Y x-\overleftarrow{\mathcal{L}}^Y\overrightarrow{\mathcal{L}}^Y x\Big)=0. \tag{98}$$

This leads us to the identity:

$$\overrightarrow{\mathcal{L}}^Y\overleftarrow{\mathcal{L}}^Y x=-\frac{1}{m}\nabla V(x). \tag{99}$$

Lemma F.9. *We have the following bound:*

$$\int_0^t \mathbb{E}^X\Big\|\overrightarrow{\mathcal{L}}^{X'}\overleftarrow{\mathcal{L}}^X X(t)+\frac{1}{m}\nabla V(X(t))\Big\|^2\mathrm{d}t\leq 2L_1(\widetilde{v},\widetilde{u})+2L_2(\widetilde{v},\widetilde{u}).$$

Proof. Consider rewriting losses as:

$$L_1(\widetilde{v},\widetilde{u})=\int_0^t \mathbb{E}_{t\sim U[0,T]}\mathbb{E}^X\Big\|\frac{1}{2}\big(\overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}X(t)+\overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}X(t)\big)+\frac{1}{m}\nabla V(X(t))\Big\|^2\mathrm{d}t, \tag{100}$$

$$L_2(\widetilde{v},\widetilde{u})=\frac{1}{4}\int_0^t \mathbb{E}_{t\sim U[0,T]}\mathbb{E}^X\Big\|\overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}X(t)-\overrightarrow{\mathcal{L}}^{X'}\overleftarrow{\mathcal{L}}^X X(t)\Big\|^2\mathrm{d}t. \tag{101}$$

Using the triangle inequality yields the statement. □

Lemma F.10. *We have the following bound:*

$$\begin{aligned}&\int_0^t \mathbb{E}^X\Big\|\overleftarrow{\mathcal{L}}^X\overrightarrow{\mathcal{L}}^X X(t)+\frac{1}{m}\nabla V(X(t))\Big\|^2\mathrm{d}t\\&\quad\leq 2T\big(\|\nabla\widetilde{u}\|_\infty+\|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\big(L_3(\widetilde{v},\widetilde{u})+L_2(\widetilde{v},\widetilde{u})\big)+4L_1(\widetilde{v},\widetilde{u})+4L_2(\widetilde{v},\widetilde{u}).\end{aligned}$$

Proof. From (76) we have:

$$\overleftarrow{\mathcal{L}}^X\overrightarrow{\mathcal{L}}^X X(t)=\overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^X X(t)+\langle\frac{\hbar}{m}\nabla\log p_X-2\widetilde{u},\nabla(\widetilde{v}+\widetilde{u})\rangle. \tag{102}$$

Noting that $\langle\frac{\hbar}{m}\nabla\log p_X-2\widetilde{u},\nabla(\widetilde{v}+\widetilde{u})\rangle\leq\big(\|\nabla\widetilde{u}\|_\infty+\|\nabla\widetilde{v}\|_\infty\big)\Big\|\frac{\hbar}{m}\nabla\log p_X-2\widetilde{u}\Big\|$ and using triangle inequality we obtain the bound:

$$\int_0^t \mathbb{E}^X\Big\|\overleftarrow{\mathcal{L}}^X\overrightarrow{\mathcal{L}}^X X(t)+\frac{1}{m}\nabla V(X(t))\Big\|^2\mathrm{d}t \tag{103}$$

$$\leq 2\big(\|\widetilde{u}\|_\infty+\|\widetilde{v}\|_\infty\big)^2\int_0^t \mathbb{E}^X\Big\|u(X(t),t)-\frac{\hbar}{2m}\log p_X(X(t),t)\Big\|^2\mathrm{d}t+4L_1(\widetilde{v},\widetilde{u})+4L_2(\widetilde{v},\widetilde{u}). \tag{104}$$

Using Theorem F.8 concludes the proof. □

Lemma F.11. *Denote $Z(t)=(X(t),Y(t))$ as compound process. For functions $h(x,y,t)=f(x,t)+g(y,t)$ we have the following identity:*

$$\overrightarrow{\mathcal{L}}^Z h=\overrightarrow{\mathcal{L}}^X f+\overrightarrow{\mathcal{L}}^Y g \tag{105}$$

Proof. A generator is a linear operator by very definition. Thus, it remains to prove only

$$\overrightarrow{\mathcal{L}}^Z f=\overrightarrow{\mathcal{L}}^X f \tag{106}$$

Since the definition of $\overrightarrow{\mathcal{F}}_t$ already contains all past events for both processes $X(t),Y(t)$, we see that this is a tautology. □

As a direct application of this Lemma, we obtain the following Corollary (by applying it twice):

Corollary F.12. *We have the following identity:*

$$\overleftarrow{\mathcal{L}}^Z\overrightarrow{\mathcal{L}}^Z\big(X(t)-Y(t)\big)=\overleftarrow{\mathcal{L}}^X\overrightarrow{\mathcal{L}}^X X(t)-\overleftarrow{\mathcal{L}}^Y\overrightarrow{\mathcal{L}}^Y Y(t).$$

Theorem F.13. *(Strong Convergence) Let the loss be defined as $\mathcal{L}(\widetilde{v},\widetilde{u})=\sum_{i=1}^4 w_iL_i(\widetilde{v},\widetilde{u})$ for some arbitrary constants $w_i>0$. Then we have the following bound between processes X and Y:*

$$\sup_{t\leq T}\mathbb{E}\|X(t)-Y(t)\|^2\leq C_T\mathcal{L}(\widetilde{v},\widetilde{u}) \tag{107}$$

where $C_T=\max_i\frac{w_i'}{w_i}$, $w_1'=4e^{T(T+1)}$, $w_2'=e^{T(T+1)}\Big(2T\big(\|\nabla\widetilde{u}\|_\infty+\|\nabla\widetilde{v}\|_\infty\big)^2e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}+4\Big)$, $w_3'=2Te^{T(T+1)}\Big(1+\big(\|\nabla\widetilde{u}\|_\infty+\|\nabla\widetilde{v}\|_\infty\big)^2e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\Big)$, $w_4'=2Te^{T(T+1)}$.

Proof. We are going to prove the bound:

$$\sup_{t\leq T}\mathbb{E}\|X(t)-Y(t)\|^2\leq\sum_{i=1}^4 w_i'L_i(\widetilde{v},\widetilde{u}) \tag{108}$$

for constants that we obtain from the Lemmas above. Then we will use the following trick to get the bound with arbitrary weights:

$$\sum_{i=1}^4 w_i'L_i(\widetilde{v},\widetilde{u})\leq\sum_{i=1}^4\frac{w_i'}{w_i}w_iL_i(\widetilde{v},\widetilde{u})\leq\big(\max_i\frac{w_i'}{w_i}\big)\sum_{i=1}^4 w_iL_i(\widetilde{v},\widetilde{u})=C_T\mathcal{L}(\widetilde{v},\widetilde{u}) \tag{109}$$

First, we apply Lemma F.5 to $Z=X-Y$ by noting that $\big[X(t)-Y(t),X(t)-Y(t)\big]_t\equiv 0$ and $\|X(0)-Y(0)\|^2=0$ almost surely:

$$\begin{aligned}
&\mathbb{E}^Z\|X(t)-Y(t)\|^2 && (110)\\
&=\int_0^t\mathbb{E}^Z\Big(2\langle\overleftarrow{\mathcal{L}}^Z(X(0)-Y(0)),X(s)-Y(s)\rangle && (111)\\
&\quad+2\int_0^s\langle\overleftarrow{\mathcal{L}}^Z\overrightarrow{\mathcal{L}}^Z(X(z)-Y(z)),X(s)-Y(s)\rangle\mathrm{d}z\Big)\mathrm{d}s && (112)\\
&\leq\int_0^t\mathbb{E}^Z\Big(\big\|\overleftarrow{\mathcal{L}}^Z(X(0)-Y(0))\big\|^2+\|X(s)-Y(s)\|^2 && (113)\\
&\quad+\int_0^s\Big(\big\|\overleftarrow{\mathcal{L}}^Z\overrightarrow{\mathcal{L}}^Z(X(z)-Y(z))\big\|^2+\|X(s)-Y(s)\|^2\mathrm{d}z\Big)\Big)\mathrm{d}s && (114)\\
&\leq\int_0^t\mathbb{E}^Z\Big(\big\|\overleftarrow{\mathcal{L}}^Z(X(0)-Y(0))\big\|^2+(1+T)\|X(s)-Y(s)\|^2 && (115)\\
&\quad+\int_0^s\big\|\overleftarrow{\mathcal{L}}^Z\overrightarrow{\mathcal{L}}^Z(X(z)-Y(z))\big\|^2\mathrm{d}z\Big)\mathrm{d}s. && (116)
\end{aligned}$$

Then, using Corollary F.12, (99) and then Lemma F.10 we obtain that

$$\begin{aligned}
\int_0^s\big\|\overleftarrow{\mathcal{L}}^Z\overrightarrow{\mathcal{L}}^Z(X(z)-Y(z))\big\|^2\mathrm{d}z&=\int_0^s\big\|\overleftarrow{\mathcal{L}}^Z\overrightarrow{\mathcal{L}}^Z X(z)+\frac{1}{m}\nabla V(X(z))\big\|^2\mathrm{d}z && (117)\\
&\leq 2T\big(\|\nabla\widetilde{u}\|_\infty+\|\nabla\widetilde{v}\|_\infty\big)^2e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\big(L_3(\widetilde{v},\widetilde{u})+L_2(\widetilde{v},\widetilde{u})\big)+4L_1(\widetilde{v},\widetilde{u})+4L_2(\widetilde{v},\widetilde{u}). && (118)
\end{aligned}$$

To deal with the remaining term involving $X(0)-Y(0)$ we observe that:

$$\int_0^t\mathbb{E}^Z\Big(\big\|\overleftarrow{\mathcal{L}}^Z(X(0)-Y(0))\big\|^2\leq 2TL_3(\widetilde{v},\widetilde{u})+2TL_4(\widetilde{v},\widetilde{u}), \tag{119}$$

where we used triangle inequality. Combining obtained bounds yields:

$$\mathbb{E}^Z\|X(t)-Y(t)\|^2 \tag{120}$$
$$\leq \int_0^t (1+T)\|X(s)-Y(s)\|^2 \mathrm{d}s \tag{121}$$
$$+ 2TL_3(\widetilde{v},\widetilde{u}) + 2TL_4(\widetilde{v},\widetilde{u}) \tag{122}$$
$$+ 2T\big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\big(L_3(\widetilde{v},\widetilde{u}) + L_2(\widetilde{v},\widetilde{u})\big) \tag{123}$$
$$+ 4L_1(\widetilde{v},\widetilde{u}) + 4L_2(\widetilde{v},\widetilde{u}) \tag{124}$$
$$= \int_0^t (1+T)\|X(s)-Y(s)\|^2 \mathrm{d}s \tag{125}$$
$$+ 4L_1(\widetilde{v},\widetilde{u}) + \Big(2T\big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T} + 4\Big)L_2(\widetilde{v},\widetilde{u}) \tag{126}$$
$$+ 2T\Big(1 + \big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\Big)L_3(\widetilde{v},\widetilde{u}) + 2TL_4(\widetilde{v},\widetilde{u}). \tag{127}$$

Finally, using integral Grönwall's inequality Gronwall (1919), we have:

$$\mathbb{E}^Z\|X(t)-Y(t)\|^2 \tag{128}$$
$$\leq 4e^{T(T+1)}L_1(\widetilde{v},\widetilde{u}) + e^{T(T+1)}\Big(2T\big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T} + 4\Big)L_2(\widetilde{v},\widetilde{u}) \tag{129}$$
$$+ 2Te^{T(T+1)}\Big(1 + \big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\Big)L_3(\widetilde{v},\widetilde{u}) + 2Te^{T(T+1)}L_4(\widetilde{v},\widetilde{u}) \tag{130}$$

□

G. Applications

G.1. Bounded Domain $\mathcal{M}$

Our approach assumes that the manifold $\mathcal{M}$ is flat or curved. For bounded domains $\mathcal{M}$, e.g., like it is assumed in PINN or any other grid-based methods, our approach can be applied if we embed $\mathcal{M} \subset \mathbb{R}^d$ and define a new family of smooth non-singular potentials V_α on entire $\mathbb{R}^d$ such that $V_\alpha \to V$ when restricted to $\mathcal{M}$ and $V_\alpha \to +\infty$ on $\partial(\mathcal{M}, \mathbb{R}^d)$ (boundary of the manifold in embedded space) as $\alpha \to 0_+$.

G.2. Singular Initial Conditions

It is possible to apply Algorithm 1 to $\psi_0 = \delta_{x_0} e^{iS_0(x)}$ for some $x_0 \in \mathcal{M}$. We need to augment the initial conditions with a parameter $\alpha > 0$ as $\psi_0 = \sqrt{\frac{1}{\sqrt{2\pi\alpha^2}} e^{-\frac{(x-x_0)^2}{2\alpha^2}}}$ for small enough $\alpha > 0$. In that case, $u_0(x) = -\frac{\hbar}{2m}\frac{(x-x_0)}{\alpha}$. We must be careful with choosing α to avoid numerical instability. It makes sense to try $\alpha \propto \frac{\hbar^2}{m^2}$ as $\frac{X(0)-x_0}{\alpha} = \mathcal{O}(\sqrt{\alpha})$. We evaluated such a setup in Appendix D.1.

G.3. Singular Potential

We should augment the potential to apply our method for simulations of the atomic nucleus with Bohr-Oppenheimer approximation (Woolley & Sutcliffe, 1977). A potential arising in this case has components of form $\frac{a_{ij}}{\|x_i - x_j\|}$. Basically, it has singularities when $x_i = x_j$. In case when x_j is fixed, our manifold is $\mathcal{M}\backslash\{x_j\}$, which has a non-trivial cohomology group.

When such potential arises, we suggest to augment the potential V_α (e.g., replace all $\frac{a_{ij}}{\|x_i - x_j\|}$ with $\frac{a_{ij}}{\sqrt{\|x_i - x_j\|^2 + \alpha}}$) so that V_α is smooth and non-singular everywhere on $\mathcal{M}$. In that case we have that $V_\alpha \to V$ as $\alpha \to 0$. With the augmented potential V_α, we can apply stochastic mechanics to obtain an equivalent to quantum mechanics theory. Of course, augmentation will produce bias, but it will be asymptotically negligent as $\alpha \to 0$.

G.4. Measurement

Even though we have full trajectories and know positions for each moment, we should carefully interpret them. This is because they are not the result of the measurement process. Instead, they represent hidden variables (and u, v represent global hidden variables – what saves us from the Bells inequalities as stochastic mechanics is non-local (Nelson, 1966)).

For a fixed $t \in [0, T]$, the distribution of $X(t)$ coincides with the distribution $\mathbf{X}(t)$ for $\mathbf{X}$ being position operator in quantum mechanics. Unfortunately, a compound distribution $(X(t), X(t'))$ for $t \neq t'$ may not correspond to the compound distribution of $(\mathbf{X}(t), \mathbf{X}(t'))$; for details see Nelson (2005). This is because each $\mathbf{X}(t)$ is a result of the *measurement process*, which causes the wave function to collapse (Derakhshani & Bacciagaluppi, 2022).

Trajectories X_i are as if we could measure $\mathbf{X}(t)$ without causing the collapse of the wave function. To use this approach for predicting some experimental results involving multiple measurements, we need to re-run our method after each measurement process with the measured state as the new initial condition. This issue is not novel for stochastic mechanics. There is the same problem in classical quantum mechanics.

This "contradiction" is resolved once we realize that $\mathbf{X}(t)$ involves measurement, and thus, if we want to calculate correlations of $(\mathbf{X}(t), \mathbf{X}(t'))$ for $t < t'$ we need to do the following:

- Run Algorithm 1 with $\psi_0, V(x, t)$ and $T = t$ to get $\widetilde{u}, \widetilde{v}$.
- Run Algorithm 2 with $\widetilde{u}, \widetilde{v}, \psi_0$ to get $\{X_{Nj}\}_{j=1}^{B}$ – B last steps from trajectories X_i of length N.
- For each X_{Nj} in the batch we need to run Algorithm 1 with $\psi_0 = \delta_{X_{Nj}}, V'(x, t') = V(x, t' + t)$ (assuming that $u_0 = 0, v_0 = 0$) and $T = t' - t$ to get $\widetilde{u}_j, \widetilde{v}_j$.
- For each X_{Nj} run Algorithm 2 with batch size $B = 1$, $\psi_0 = \delta_{X_{Nj}}, \widetilde{u}_j, \widetilde{v}_j$ to get X'_{Nj}.
- Output pairs $\left\{(X_{N,j}, X'_{N,j})\right\}_{j=1}^{B}$.

Then the distribution of $(X_{N,j}, X'_{N,j})$ will correspond to the distribution of $(\mathbf{X}(t), \mathbf{X}(t'))$. This is well described and proven in Derakhshani & Bacciagaluppi (2022). Therefore, it is possible to simulate the right correlations in time using our approach, though it may require learning $2(B + 1)$ models. The promising direction of future research is to consider X_0 as a feature for the third step here and, thus, learn only $2 + 2$ models.

G.5. Observables

To estimate any scalar observable of form $\mathbf{Y}(t) = y(\mathbf{X}(t))$ in classic quantum mechanics one needs to calculate:

$$\langle \mathbf{Y} \rangle_t = \int_{\mathcal{M}} \overline{\psi(x, t)} y(x) \psi(x, t) \mathrm{d}x.$$

In our setup, we can calculate this using the samples $X_{\left[\frac{Nt}{T}\right]} \approx X(t) \sim \left|\psi(\cdot, t)\right|^2$:

$$\langle \mathbf{Y} \rangle_t \approx \frac{1}{B} \sum_{j}^{B} y(X_{\left[\frac{Nt}{T}\right]_j}),$$

where $B \geq 1$ is the batch size, N is the time discretization size. The estimation error has magnitude $\mathcal{O}(\frac{1}{\sqrt{B}} + \epsilon + \varepsilon)$, where $\epsilon = \frac{T}{N}$ and ε is the L_2 error of recovering true u, v. In our paper, we have not bounded ε but provide estimates for it in our experiments against the finite difference solution.[9]

G.6. Wave Function

Recovering the wave function from u, v is possible using a relatively slow procedure. Our experiments do not cover this because our approach's main idea is to avoid calculating the wave function. But for the record, it is possible. Assume we

[9] If we are able to reach $\mathcal{L}(\theta) = 0$ then essentially $\varepsilon = 0$. We leave bounding ε by $\mathcal{L}(\theta_\tau)$ for future work.

solved equations for u, v. We can get the phase and density by integrating Equation (20):

$$S(x,t) = S(x,0) + \int_0^t \Big(\frac{1}{2m}\langle \nabla, u(x,t)\rangle + \frac{1}{2\hbar}\big\|u(x,t)\big\|^2 - \frac{1}{2\hbar}\big\|v(x,t)\big\|^2 - \frac{1}{\hbar}V(x,t)\Big)\mathrm{d}t, \tag{131}$$

$$\rho(x,t) = \rho_0(x)\exp\Big(\int_0^t \big(-\langle \nabla, v(x,t)\rangle - \frac{2m}{\hbar}\langle u(x,t), v(x,t)\rangle\big)\Big)\mathrm{d}t \tag{132}$$

This allows us to define $\psi = \sqrt{\rho(x,t)}e^{iS(x,t)}$, which satisfies the Schrödinger equation (1). Suppose we want to estimate it over a grid with N time intervals and $\lceil\sqrt{N}\rceil$ intervals for each coordinate (a typical recommendation for Equation (1) is to have a grid satisfying $\mathrm{d}x^2 \approx \mathrm{d}t$). It leads to a sample complexity of $\mathcal{O}(N^{\frac{d}{2}+1})$, which is as slow as other grid-based methods for quantum mechanics. The error in that case will also be $\mathcal{O}(\sqrt{\epsilon} + \varepsilon)$ (Smith & Smith, 1985).

H. On Criticism of Stochastic Mechanics

Three major concerns arise regarding stochastic mechanics developed by Nelson (1966); Guerra (1995):

- The proof of the equivalence of stochastic mechanics to classic quantum mechanics relies on an implicit assumption of the phase $S(x,t)$ being single-valued (Wallstrom, 1989).
- If there is an underlying stochastic process of quantum mechanics, it should be non-Markovian (Nelson, 2005).
- For a quantum observable, e.g., a position operator $\mathbf{X}(t)$, a compound distribution of positions at two different timestamps t, t' does not match the distribution of $(\mathbf{X}(t), \mathbf{X}(t'))$ (Nelson, 2005).

Appendix G.4 discusses why a mismatch of the distributions is not a problem and how we can adopt stochastic mechanics with our approach to get correct compound distributions by incorporating the measurement process into the stochastic mechanical picture.

H.1. On "Inequivalence" to Schrödinger Equation

This problem is explored in the paper by Wallstrom (1989). Firstly, the authors argue that proofs of the equivalency in Nelson (1966); Guerra (1995) are based on the assumption that the wave function phase S is single-valued. In the general case of a multi-valued phase, the wave functions are identified with sections of complex line bundles over $\mathcal{M}$. In the case of a trivial line bundle, the space of sections can be formed from single-valued functions, see Alvarez (1986). The equivalence class of line bundles over a manifold $\mathcal{M}$ is called Picard group, and for smooth manifolds, $\mathcal{M}$ is isomorphic to $H^2(\mathcal{M}, \mathbb{Z})$, so-called second cohomology group over $\mathbb{Z}$, see Prieto & Vitolo (2014) for details. Elements in this group give rise to non-equivalent quantizations with irremovable gauge symmetry phase factor.

Therefore, *in this paper, we assume that* $H^2(\mathcal{M}, \mathbb{Z}) = 0$, which allows us to eliminate all criticism about non-equivalence. Under this assumption, stochastic mechanics is *equivalent* indeed. This condition holds when $\mathcal{M} = \mathbb{R}^d$. Though, if a potential V has singularities, e.g., $\frac{a}{\|x - x_*\|}$, then we should exclude x_* from $\mathbb{R}^d$ which leads to $\mathcal{M} = \mathbb{R}^d \backslash \{x_*\}$ and this manifold satisfies $H^2(\mathcal{M}, \mathbb{Z}) \cong \mathbb{Z}$ (May, 1999), which essentially leads to "counterexample" provided in Wallstrom (1989). We suggest a solution to this issue in Appendix G.2.

H.2. On "Superluminal" Propagation of Signals

We want to clarify why this work should not be judged from perspectives of *physical realism*, *correspondence to reality* and *interpretations* of quantum mechanics. This tool gives the exact predictions as classical quantum mechanics at a moment of measurement. Thus, we do not care about a superluminal change in the drifts of entangled particles and other problems of the Markovian version of stochastic mechanics.

H.3. Non-Markovianity

Nelson believes that an underlying stochastic process of reality should be non-Markovian to avoid issues with the Markovian processes like superluminal propagation of signals (Nelson, 2005). Even if such a process were proposed in the future, it would not affect our approach. In stochastic calculus, there is a beautiful theorem from Gyöngy (1986):

Theorem H.1. *Assume $X(t), F(t), G(t)$ are adapted to Wiener process $W(t)$ and satisfy:*

$$\mathrm{d}X(t) = F(t)\mathrm{d}t + G(t)\mathrm{d}\vec{W}.$$

Then there exist a Markovian process $Y(t)$ satisfying

$$\mathrm{d}Y(t) = f(Y(t), t)\mathrm{d}t + g(Y(t), t)\mathrm{d}\vec{W}$$

where $f(x,t) = \mathbb{E}(F(t)\|X(t) = x)$,$g(x,t) = \sqrt{\mathbb{E}(G(t)G(t)^T\|X(t) = x)}$ and such that $\forall t$ holds $\mathrm{Law}(X(t)) = \mathrm{Law}(Y(t))$.

This theorem tells us that we already know how to build a process $Y(t)$ without knowing $X(t)$; it is stochastic mechanics by Nelson (Guerra, 1995; Nelson, 1966) that we know. From a numerical perspective, we better stick with $Y(t)$ as it is easier to simulate, and as we explained, we do not care about correspondence to reality as long as it gives the same final results.

H.4. Ground State

Unfortunately, our approach is unsuited for the ground state estimation or any other stationary state. FermiNet (Pfau et al., 2020) does a fantastic job already. The main focus of our work is time evolution. It is possible to estimate some observable $\mathbf{Y}$ for the ground state if its energy level is unique and significantly lower than others. In that case, the following value approximately equals the group state observable for $T \gg 1$:

$$\langle \mathbf{Y} \rangle_{ground} \approx \frac{1}{T}\int_0^T \langle \mathbf{Y} \rangle_t \mathrm{d}t \approx \frac{1}{NB}\sum_{i=1}^{N}\sum_{j=1}^{B} y(X_{ij})$$

This works only if the ground state is unique, and the initial conditions satisfy $\int_{\mathcal{M}} \overline{\psi_0}\psi_{ground}\mathrm{d}x \neq 0$, and its energy is well separated from other energy levels. In that scenario, oscillations will cancel each other out.

I. Future Work

This section discusses possible directions for future research. Our method is a promising direction for fast quantum mechanics simulations, but we consider the most straightforward setup in our work. Possible future advances include:

- In our work, we consider the simplest integrator of SDE (Euler-Maruyama), which may require setting $N \gg 1$ to achieve the desired accuracy. However, a higher-order integrator (Smith & Smith, 1985) or an adaptive integrator (Ilie et al., 2015) should achieve the desired accuracy with much lower N.
- Exploring the applicability of our method to fermionic systems is a promising avenue for future investigation. Successful extensions in this direction would not only broaden the scope of our approach but also have implications for designing novel materials, optimizing catalytic processes, and advancing quantum computing technologies.
- It should be possible to extend our approach to a wide variety of other quantum mechanical equations, including Dirac and Klein-Gordon equations used to account for special relativity (Serva, 1988; Blanchard et al., 2005), a non-linear Schrödinger equation (1) used in condensed matter physics (Serkin & Hasegawa, 2000) by using McKean-Vlasov SDEs and the mean-field limit (Buckdahn et al., 2017; dos Reis et al., 2022), and the Shrödinger equation with a spin component (Dankel, 1970; De Angelis et al., 1991).
- We consider a rather simple, fully connected architecture of neural networks with $\tanh$ activation and three layers. It might be more beneficial to consider specialized architectures for quantum mechanical simulations, e.g., Pfau et al. (2020). Permutation invariance can be ensured using a self-attention mechanism (Vaswani et al., 2017), which could potentially offer significant enhancements to model performance. Additionally, incorporating gradient flow techniques as suggested by Neklyudov et al. (2024) can help to accelerate our algorithm.
- Many practical tasks require knowledge of the error magnitude. Thus, providing explicit bounds on ε in terms of $\mathcal{L}(\theta_M)$ is critical.