THE UNIVERSITY OF CHICAGO

MACHINE LEARNING FOR COMPLEX PHYSICAL SYSTEMS: APPLICATIONS TO CLIMATE FORECASTING AND DYNAMICAL SYSTEMS SIMULATION

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY
ELENA ORLOVA

CHICAGO, ILLINOIS
AUGUST 2025

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This dissertation investigates machine learning (ML) approaches for modeling complex physical systems across multiple domains. We examine three interconnected research areas: subseasonal climate forecasting, chaotic dynamical systems, and quantum mechanical simulations. Across these diverse domains, we address common challenges in the application of ML to physical systems: effectively encoding information to capture important features of spatiotemporal dynamics, developing robust emulators of physical systems that preserve critical invariant properties such as chaotic attractors, and systematic incorporation of known physical constraints into neural architectures and training schemes to simultaneously enhance prediction accuracy and computational efficiency. These methodological considerations form the conceptual framework for our various applications. For subseasonal climate forecasting, we develop frameworks that use lagged ensemble members and observational data to significantly improve temperature and precipitation forecasts. In chaotic dynamical systems, we propose neural operator training approaches that preserve invariant measures through optimal transport distance minimization and contrastive learning, maintaining statistical fidelity in long-term simulations. For quantum mechanics, we introduce Deep Stochastic Mechanics (DSM), a framework inspired by stochastic mechanics and generative diffusion models that may have far lower computational complexity in higher dimensions compared to traditional numerical methods by exploiting wave function latent structure. Our DSM simulations of bosonic systems outperform conventional approaches in both accuracy and computational efficiency. Furthermore, we extend DSM to effectively model fermionic systems, demonstrating its capabilities through a hydrogen molecule time dynamics simulation. Throughout these applications, we integrate domain-specific physics with advanced learning techniques to enable more accurate, efficient, and physically consistent simulations.

PREFACE

This thesis is the result of a few collaborations. Chapter 2 is based on the publication "Beyond ensemble averages: Leveraging climate model ensembles for subseasonal forecasting," published in the Artificial Intelligence for the Earth Systems journal in April 2024 [Orlova et al., 2024a] and jointly authored with Rebecca Willett, Haokun Liu, Raphael Rossellini at the University of Chicago; Benjamin A. Cash at the George Mason University. This work was supported by the NSF (OAC-1934637, DMS-1930049, and DMS-2023109) and C3.ai. I (E. Orlova) developed and implemented the algorithms, conducted experiments, and generated tables and figures for both regression and tercile classification tasks for precipitation and temperature. I authored the majority of the manuscript, particularly the sections on the aforementioned tasks and most analysis components. I collaborated with H. Liu on the U-Net implementation, while H. Liu primarily managed data acquisition and preprocessing with my assistance. H. Liu conducted experiments for the quatile regression task. R. Rossellini contributed the random forest implementation for regression and quantile tasks, and provided substantial manuscript editing support. B. Cash contributed the climatological context and interpretation in the manuscript.

Chapter 3 is drawn, with minor modifications, from a manuscript "Training neural operators to preserve invariant measures of chaotic attractors" accepted at the NeurIPS 2023 conference, and published in Advances in Neural Information Processing Systems [Jiang et al., 2023]. This work is jointly authored with Rebecca Willett along with Ruoxie Jiang and Peter Y. Lu at at the University of Chicago. This work was supported by DOE grant DE-SC0022232, AFOSR grant FA9550-18-1-0166, and NSF grants DMS-2023109 and DMS-1925101. In this collaboration, I (E. Orlova) contributed by conducting a few additional experiments, verifying experimental results, refining the codebase, and participating in research discussions, while R. Jiang led the development and implementation of all algorithms and the design and execution of all experiments. P. Lu conducted the theoretical analysis that underpins this work.

Chapter 4 is primarily drawn from from a manuscript “Deep Stochastic Mechanics” accepted at the ICML 2024 conference, and published in Proceedings of the 41st International Conference on Machine Learning [Orlova et al., 2024b]. This work is jointly authored with Aleksei Ustimenko from Simulacra AI along with Rebecca Willett, Ruoxie Jiang and Peter Y. Lu at at the University of Chicago. This project was supported by DOE DE-SC0022232, NSF DMS-2023109, NSF PHY2317138, NSF 2209892, and the University of Chicago Data Science Institute. I (E. Orlova) implemented the algorithms, conducted all the experimental studies, and produced the tables and figures presented in the paper. Aleksei Ustimenko conceived the central idea for our proposed methodology and developed its theoretical framework; he provided crucial guidance during algorithm implementation and experimental design. We jointly authored most of the manuscript through extensive collaboration. R. Jiang contributed by executing additional experiments and participating in research discussions, while P. Lu provided valuable insights for the theoretical and practical aspects of the work.

We currently work on an extension of this stochastic framework to fermionic systems, called Fermionic Depp Stochastic Mechanics (F-DSM), which is a manuscript in preparation with the same team of collaborators. This work represents a significant theoretical advancement in applying stochastic mechanics principles to quantum systems with antisymmetry, extending the original methodology to work on a broader class of systems. Chapter 5 describes our F-DSM framework.

CHAPTER 1
INTRODUCTION

The simulation and prediction of complex physical systems represent fundamental challenges across scientific disciplines. For centuries, scientists have developed increasingly sophisticated mathematical frameworks to describe such phenomena, resulting in powerful theories that have advanced our understanding of physics, chemistry, and earth sciences. However, despite these theoretical achievements, many complex physical systems remain challenging to simulate and predict with desired accuracy, especially when computational resources are limited or when the underlying dynamics exhibit chaotic or high-dimensional behaviors.

The last decade has witnessed a paradigm shift in computational approaches to studying complex physical systems. Machine learning, once primarily used in computer science and data analytics, now complements traditional numerical methods in scientific computing [Brunton and Kutz, 2022]. This intersection has led to the rapidly evolving field of scientific machine learning [Roscher et al., 2020, Carleo et al., 2019], which seeks to leverage data-driven approaches while respecting physical constraints, conservation laws, and domain-specific knowledge. Rather than replacing first-principles approaches, ML offers new ways to enhance simulation capabilities, discover patterns in high-dimensional data, accelerate computations, and reveal insights that might otherwise remain hidden.

Nonetheless, the application of ML to the natural sciences presents significant challenges. Physical systems must obey fundamental laws, conserve certain quantities, and respect invariances that purely data-driven approaches might violate. Scientific applications often require interpretability and uncertainty quantification – aspects that are challenging for complex ML models. Across diverse domains, from climate systems to quantum mechanics, these critical challenges consistently emerge when applying ML techniques: (1) Effective information encoding presents the challenge of powerful and effetive represention of complex spatiotemporal dynamics. Whether capturing geographic dependencies in climate patterns or the evolution of quantum wave functions, developing efficient representations that capture these features fundamental to successful modeling. (2) Preserving invariant properties is crucial for physical fidelity, particularly in long-term simulations. While traditional ML approaches focus on minimizing immediate prediction errors, they often fail to maintain long-term time-invariant statistics, conservation laws, and other invariants that characterize system's behavior. (3) Incorporating physical constraints can systematically improve both accuracy and efficiency. Rather than relying on purely data-driven approaches, integrating domain knowledge, whether through physics-based ensemble predictions in climate forecasting or quantum mechanical equations in particle simulations, provides critical guidance for model development.

The collection of works presented in this dissertation makes several substantive contributions to the growing field of scientific machine learning. By combining physical insight with ML techniques, we achieve improved accuracy, better preservation of essential physical properties, and potentially significant computational speedups, advancing the frontiers of what can be simulated and predicted in complex physical systems.

1.1 Subseasonal climate forecasting

Climate science is a prime example of a domain where ML approaches prove increasingly valuable [Reichstein et al., 2019]. The Earth's climate system involves complex interactions across multiple scales, from microscale cloud physics to global atmospheric circulation patterns, making accurate forecasting a challenging problem. Traditional numerical weather prediction models solve discretized versions of the governing fluid dynamics equations, often demanding massive computational resources and facing fundamental limits in their predictive horizon. ML methods offer alternative approaches that can complement these physics-based models by learning from historical data, identifying subtle patterns, and potentially extending the range of skillful forecasts, particularly at the challenging subseasonal-to-seasonal timescales that have significant societal importance.

In our work on subseasonal forecasting [Orlova et al., 2024a], we address this critical gap by developing ML approaches that leverage ensemble forecasts in novel ways. Unlike previous methods that typically use only the ensemble mean in addition to observable data, we demonstrate that the ensemble members contain valuable information beyond a simple mean. Our approach incorporates lagged numerical ensemble forecasts (where members have different initialization dates) along with observational data including relative humidity, pressure at sea level, and geopotential height. We employ various ML methods, including linear models, random forests, convolutional neural networks, and stacked models, to predict monthly average precipitation and temperature two weeks in advance for the continental United States. Our results show significant improvements over standard baselines such as climatological forecasts and ensemble means, with particular success in extreme event predictions, crucial for planning and mitigation efforts. Chapter 2 provides more details.

1.2 Emulating chaotic dynamical systems

Simulation of dynamical systems, particularly those exhibiting chaotic behavior, presents unique challenges that ML approaches can help address. Chaotic systems are characterized by their sensitive dependence on initial conditions, making long-term trajectory prediction difficult. However, many applications do not require exact trajectories but rather statistical properties or invariant measures that characterize the system's long-term behavior.

In our work on chaotic systems [Jiang et al., 2023], we demonstrate that mean-squared-error (MSE) objectives, especially under noisy conditions, collapse trajectories into "averaged" dynamics lacking high-frequency features and critical physical signatures (e.g., Lyapunov exponents). To overcome this, we propose a framework designed to preserve invariant measures of chaotic attractors that characterize the time-invariant statistical properties of the dynamics. In particular, we introduce two novel approaches: a loss based on optimal transport distance between observed dynamics and neural operator outputs (requiring expert knowledge of underlying physics), and a contrastive learning framework that preserves statistical properties without specialized prior knowledge. These losses are combined with standard MSE loss evaluated over short time horizons, ensuring short-term predictability while properly reproducing long-term statistical properties. Our method is empirically shown to preserve invariant measures of chaotic attractors across various systems, offering a principled approach that extends beyond traditional accuracy metrics to capture the essential statistical character of complex physical systems. Chapter 3 provides more details.

1.3 Quantum mechanics dynamics simulations

1.3.1 Bosonic systems

At the atomic and molecular scale, quantum mechanical systems behave fundamentally different from classical systems. The curse of dimensionality, where computational complexity grows exponentially with system size, limits direct simulation approaches to relatively small quantum systems. Recent advances in ML, particularly in generative modeling and representation learning [Carleo and Troyer, 2017, Hermann et al., 2023], open new possibilities for approximating quantum states and dynamics in ways that may circumvent the curse of dimensionality by exploiting the latent structure of physically relevant quantum states. The time-dependent Schrd̈inger equation (TDSE), which governs quantum evolution, exemplifies this challenge with applications that span quantum chemistry, drug discovery, condensed matter physics, and quantum computing.

Our work introduces Deep Stochastic Mechanics (DSM) [Orlova et al., 2024b], a novel deep learning framework that addresses these computational limitations through a stochastic interpretation of quantum mechanics. Rather than directly solving the Schrd̈inger equation, DSM learns stochastic processes whose marginal distributions match the quantum mechanical probability density [Nelson, 1966b]. Inspired by the success of deep diffusion models in generative modeling [Croitoru et al., 2023], our approach adapts to the latent low-dimensional structure of wave functions, potentially offering computational complexity that scales quadratically rather than exponentially with dimension. We develop novel equations for stochastic quantum mechanics that require only first-order derivatives, resulting in improved computational efficiency. Our theoretical analysis establishes strong convergence guarantees, while numerical experiments verify the method's advantages for both non-interacting and interacting bosonic systems. Chapter 4 provides more details.

1.3.2 Extension to fermionic systems

Building on the DSM framework, we extend our stochastic mechanics approach to fermionic systems, which are fundamental to chemistry and materials science. Fermionic wave functions must satisfy antisymmetry properties arising from the Pauli exclusion principle, which prohibits identical fermions from occupying the same quantum state. We introduce Fermionic DSM (F-DSM), which incorporates these essential quantum mechanical constraints through specialized wave function representations and modified training scheme. Our implementation reasonably simulates dynamics for a hydrogen molecule, bridging our theoretical framework with practical quantum chemical applications. This extension proves that stochastic-mechanics-based approaches can handle real molecular systems, opening pathways for applying these methods to larger chemical systems, and potentially advancing computational drug discovery and materials design. Chapter 5 provides more details on the F-DSM framework.

CHAPTER 2

BEYOND ENSEMBLE AVERAGES: LEVERAGING CLIMATE MODEL ENSEMBLES FOR SUBSEASONAL FORECASTING

2.1 Introduction

High-quality forecasts of key climate variables such as temperature and precipitation on subseasonal time scales, defined here as the time range between two weeks and two months, have long been a gap in operational forecasting [Ban et al., 2016]. Advances in weather forecasting on time scales from days to about a week [Lorenc, 1986, National Academies of Sciences, 2016, National Research Council, 2010, Simmons and Hollingsworth, 2002] or seasonal forecasts on time scales of two to nine months [Barnston et al., 2012] do not necessarily translate to the challenging subseasonal regime. Addressing the crucial need for forecasts on the seasonal-to-subseasonal (S2S) timescale, collaborative initiatives led by the World Weather Research Programme and the World Climate Research Programme aim to advance S2S forecasting by focusing on mesoscale–planetary-scale interactions, high-resolution simulations, data assimilation methods, and tailored socioeconomic support [Brunet et al., 2010]. Skillful forecasts on subseasonal time scales would have immense value in agriculture, insurance, and economics [White et al., 2022, Mouatadid et al., 2023]. The importance of improved subseasonal predictions is also detailed by Ban et al. [2016] and National Research Council [2010].

The National Centers for Environmental Prediction (NCEP), part of the National Oceanic and Atmospheric Administration (NOAA), currently issues a “week 3-4 outlook" for the contiguous United States (CONUS).[1] The NCEP outlooks are constructed using a combination of dynamical and statistical forecasts, with statistical forecasts based largely on how conditions in the past have varied (linearly) with indices of the El Niño-Southern Oscillation

1. https://www.cpc.ncep.noaa.gov/products/predictions/WK34/

(ENSO), Madden-Julian Oscillation (MJO), and global warming (i.e., the 30-year trend). There exists great potential to advance subseasonal forecasting (SSF) using machine learning (ML) techniques. Haupt et al. [2021] provides an overview of using ML methods for post-processing of numerical weather predictions. Vannitsem et al. [2021] highlight the crucial role of statistical post-processing techniques, including ML methods, in national meteorological services. They discuss theoretical developments and operational applications, current challenges, and potential future directions, particularly focusing on translating research findings into operational practices. A real-time forecasting competition called the Subseasonal Climate Forecast Rodeo [Hwang et al., 2018], sponsored by the Bureau of Reclamation in partnership with NOAA, USGS, and the U.S. Army Corps of Engineers, illustrated that teams using ML techniques can outperform forecasts from NOAA's operational seasonal forecast system.

Here, we present work focused on developing ML-based forecasts that leverage lagged ensembles (i.e., an ensemble whose members are initialized from a succession of different start dates) of forecasts produced by NCEP in addition to observed data and other features. Previous studies, including successful methods in the Rodeo competition (e.g., Hwang et al. [2019]), incorporate the ensemble mean as a feature in their ML systems but do not use any other information about the ensemble. In other words, variations among the ensemble members are not reflected in the training data or incorporated into the learned model. In contrast, this work *demonstrates that the full ensemble contains important information for subseasonal forecasting outside the ensemble mean.* Specifically, we consider the test case of predicting monthly 2-meter temperatures and precipitation two weeks in advance over 3000 locations over the continental United States using physics-based predictions, such as NCEP-CFSv2 hindcasts [Kirtman et al., 2014, Saha et al., 2014], using an ensemble of 24 distinct forecasts. We repeat this experiment for the Global Modeling and Assimilation Office from the National Aeronautics and Space Administration (NASA-GMAO) ensemble, which has

11 ensemble members [Nakada et al., 2018].

2.1.1 Contributions

In this context, this work makes the following contributions:

- We train a variety of ML models (including neural networks, random forests, linear regression, and model stacking) that input all ensemble member predictions as features in addition to observations of geopotential heights, relative humidity, precipitation, and temperature from past months to produce new forecasts with higher accuracy than the ensemble mean; forecast accuracy is measured with a variety of metrics (Section 2.6). These models are considered in the context of regression, quantile regression, and tercile classification. Systematic experiments are used to characterize the influence of individual ensemble members on predictive skill (Section 2.7.1).

- The collection of ML models employed allows us to consider different modes of accounting for spatial variability. ML models can account for spatial correlations among both features and targets; for example, when predicting Chicago precipitation, our models can leverage not only information about Chicago but also about neighboring regions. Specifically, we consider the following learning frameworks: (a) training a predictive model for each spatial location independently; (b) training a predictive model that inputs the spatial location as a feature and hence can be applied to any single spatial location; (c) training a predictive model for the full spatial map of temperature or precipitation – i.e., predicting an outcome for all spatial locations simultaneously. ML models present various ways to account for spatial variability, each with distinct advantages and disadvantages. Our application of model stacking (an ML technique where multiple models are combined, with their predictions used as input features for another model that produces the final prediction) allows our final learned model to exploit the advantages of each method.

- We conduct a series of experiments to help explain the learned model and which features the model uses most to make its predictions. We systematically explore the impact of using lagged observational data in addition to ensemble forecasts and positional encoding to account for spatial variations (Section 2.7.3).
- The ensemble of forecasts from a physics-based model (e.g., NCEP-CFSv2 or NASA-GMAO) contain information salient to precipitation and temperature forecasting besides their mean, and ML models that leverage the full ensemble generally outperform methods that rely on the ensemble mean alone (Section 2.7.1).
- Finally, we emphasize that the final validation of our approach was conducted on data from 2011 to 2020 that was not used during any of the training, model development, parameter tuning, or model selection steps. We only conducted our final assessment of the predictive skill for 2011 to 2020 after we had completed all other aspects of this work. Because of this, our final empirical results accurately reflect the anticipated performance of our methods on new data.

2.1.2 Related work

While statistical models were common for weather prediction in the early days of weather forecasting [Nebeker, 1995], forecasts using physics-based dynamic system models have been carried out since the 1980s and have been the dominant forecasting method in climate prediction centers since the 1990s [Barnston et al., 2012]. Many physics-based forecast models are used both in academic research and operationally. Such systems often produce ensembles of forecasts – e.g., the result of running a physics-based simulation multiple times with different initial conditions or parameters, and are a mainstay of operational forecast centers around the globe.

Recently, skillful ML approaches have been developed for short-range weather prediction

[Chen et al., 2023, Nagaraj and Kumar, 2023, Frnda et al., 2022, Herman and Schumacher, 2018, Ghaderi et al., 2017, Grover et al., 2015, Radhika and Shashi, 2009, Cofino et al., 2002] and longer-term weather forecasting [Lam et al., 2023, Yang et al., 2023a, Chen et al., 2023, Hewage et al., 2021, Cohen et al., 2019, Totz et al., 2017, Iglesias et al., 2015, Badr et al., 2014]. However, forecasting on the subseasonal timescale, with 2-8 week outlooks, has been considered a far more difficult task than seasonal forecasting due to its complex dependence on both local weather and global climate variables [Vitart et al., 2012, Min et al., 2020]. Seasonal prediction also benefits from targeting a much larger averaging period.

Some ML algorithms for subseasonal forecasting use purely observational data (i.e., not using any physics-based ensemble forecasts). He et al. [2020b] focuses on analyzing different ML methods, including Gradient Boosting trees and Deep Learning (DL) for SSF. They propose a careful construction of feature representations of observational data and show that ML methods are able to outperform a climatology baseline, i.e., predictions corresponding to the 30-year mean at a given location and time. This conclusion is based on comparing the relative R^2 scores for the ML approaches and climatology. Srinivasan et al. [2021] proposes a Bayesian regression model that exploits spatial smoothness in the data.

Other works use the ensemble mean as a feature in their ML models. For example, in the subseasonal forecasting Rodeo [Hwang et al., 2018], a prediction challenge for temperature and precipitation at weeks 3-4 and 5-6 in the western U.S. sponsored by NOAA and the U.S. Bureau of Reclamation, simple yet thoughtful statistical models consistently outperform NOAA's dynamical systems forecasts. In particular, the winning approach uses a stacked model from two nonlinear regression models, a selection of climate variables such as temperature, precipitation, sea surface temperature, sea ice concentration, and a collection of physics-based forecast models including the ensemble mean from various modeling centers in the North American Multi-Model Ensemble (NMME). From the local linear regression with multitask feature selection model analysis, the ensemble mean is the first- or second-most important feature for forecasting, especially for precipitation. He et al. [2021] perform a comparison of modern ML models that use data from the Subseasonal Experiment (SubX) project for SSF in the western contiguous United States. The experiments show that incorporating the ensemble mean as an input feature to ML models leads to a significant improvement in forecasting performance, but that work does not explore the potential value of individual ensemble members aside from the ensemble mean. Grönquist et al. [2020] note that physics-based ensembles are computationally demanding to produce and propose an ML method that can input a subset of ensemble forecasts and generate an estimate of the full ensemble; they observe that the output ensemble estimate has more prediction skill than the original ensemble. Loken et al. [2022] analyze the forecast skill of random forests leveraging the ensemble members for next-day severe weather prediction compared to only using the ensemble mean. However, their results only cover forecasts with a lead time of up to 48 hours, so it is unclear if their methods would have succeeded in the tougher subseasonal forecasting setting.

This paper complements the prior work above by developing powerful learning-based approaches that incorporate both physics-based forecast models and observational data to improve SSF over CONUS.

2.2 Data

Table 2.1 describes variables used in the experiments. Climatological means of precipitation and temperature are calculated using 1971-2000 NOAA data [NOAA, 2022]. There are many ensembles of physics-based predictions produced by forecasting systems. NMME is a collection of physics-based forecast models from various modeling centers in North America, including NOAA/NCEP, NOAA/Geophysical Fluid Dynamics Laboratory (GFDL), International Research Institute for Climate and Society (IRI), National Center for Atmospheric Research (NCAR), NASA, and Canadian Meteorological Centre [Kirtman et al., 2014]. NMME provides forecasts from multiple global forecast models from North American modeling centers [Kirtman et al., 2014]. The NMME project has two predictive periods: hindcast and forecast. A hindcast period refers to when a dynamic model re-forecasts historical events, which can help climate scientists develop and test new models to improve forecasting and to evaluate model biases. In contrast, a forecast period has real-time predictions generated from dynamic models.

Type	Variable	Description	Unit	Spatial Coverage	Time Range	Data Source
Feature variable	tmp2m	Daily average temperature at 2 meters	$^{\circ}C$	US mainland $0.5^{\circ} \times 0.5^{\circ}$ grid	1985 to 2020	CPC Global Daily Temperature [Fan and Van den Dool, 2008]
	precip	Daily average precipitation	mm	US mainland $0.5^{\circ} \times 0.5^{\circ}$ grid	1985 to 2020	CPC Global Daily Precipitation [Xie et al., 2010]
	SSTs	Daily sea surface temperature	$^{\circ}C$	Ocean only $0.25^{\circ} \times 0.25^{\circ}$ grid	1985 to 2020	Optimum Interpolation SSTs High Resolution (OISST) [Reynolds et al., 2007]
	rhum	Daily relative humidity near the surface	Pa	US mainland and North Pacific & Atlantic Ocean $0.5^{\circ} \times 0.5^{\circ}$ grid	1985 to 2020	Atmospheric Research Reanalysis Dataset [Kalnay et al., 1996]
	slp	Daily pressure at sea level	%			
	hgt500	Daily geopotential height at 500mb	m			
Climatology	tmp2m	Daily average temperature at 2 meters	K	Globally $1^{\circ} \times 1^{\circ}$ grid	1971 to 2000	NOAA [NOAA, 2022]
	precip	Daily average precipitation	mm	Globally $1^{\circ} \times 1^{\circ}$ grid	1971 to 2000	NOAA [NOAA, 2022]

Table 2.1: Description of climate variables and their data sources. Our target climate variables for sub-seasonal forecasting are precipitation and 2-meter temperature. We use NOAA data to calculate the climatology from 1971 to 2000. We also perform linear spatial interpolation on the historical values to get values with the same resolution and support as target climate variables.

We use ensemble forecasts from the NMME's NCEP-Climate Forecast System version 2 (CFSv2, Kirtman et al. [2014], Saha et al. [2014]), which has $K = 24$ ensemble members at a $1^{\circ} \times 1^{\circ}$ resolution over a 2-week lead time. NCEP-CFSv2 is the operational prediction model currently used by the U.S. Climate Prediction Center. The NCEP-CFSv2 model has two different products available in the NMME archive: we use its hindcasts from 1982 to 2010 for training and validation of our models, and we use its forecasts from April 2011 to

December 2020 for the final evaluation of our models.

In order to ensure our results are not unique to a single forecasting model, we also analyze output from the NASA-Global Modeling and Assimilation (GMAO) from the Goddard Earth Observing System model version 5 (GEOS, Nakada et al. [2018]), which has $K = 11$ ensemble members at a $1^\circ \times 1^\circ$ resolution over a 2-week lead time. Similarly, we use its hindcasts from 1981 to 2010 for training and validation of our models, and we use its forecasts from January 2011 to January 2018 for final evaluation. The test periods of NCEP-CFSv2 and NASA-GMAO data differ due to data availability. Note that the identical version of each model is used to generate the test, train, and validation data.

Different ensemble members correspond to different initial conditions of the underlying physical model. The NCEP-CFSv2 forecasts are initialized in the following way: four initializations at times 0000, 0600, 1200, and 1800 UTC every fifth day, starting one month prior to the lead time of two weeks (Table B1 in Saha et al. [2014]). NASA-GMAO is a fully coupled atmosphere–ocean–land–sea ice model, with five forecasts initialized every five days. While additional members are generated through perturbation methods closest to the beginning of each month [2], not all members are initialized on different dates, meaning that the ensemble is not strictly lagged. However, NASA-GMAO members are not interchangeable, as each is created using a distinct method.

All data are interpolated to lie on the same $1^\circ \times 1^\circ$ grid, resulting in $L = 3,274$ U.S. locations. Climate variables available daily (such as pressure at sea level or precipitation) are converted to monthly average values. When data are available as monthly averages only, we ensure that our forecast for time $t + \delta_t$ does not use any information from the interval $(t, t + \delta_t)$.

2. https://gmao.gsfc.nasa.gov/products/climateforecasts/GEOS5/DESC/init.php

2.3 Problem statement: forecasting tasks

The learning task can be formulated as learning a model $f_\theta : \mathbf{X} \to \mathbf{y}$ with parameters $\boldsymbol{\theta}$. This model f_θ can be a linear regression (where $\boldsymbol{\theta}$ is a set of regression weights), the mean of ensemble members (no $\boldsymbol{\theta}$ needs to be learned), a random forest (where $\boldsymbol{\theta}$ parameterizes the set of trees in the forest), a convolutional neural network (where $\boldsymbol{\theta}$ is the collection of neural network weights), or other learned models. We consider three forecasting tasks: regression, tercile classification, and quantile regression.

Regression The goal of regression is to predict monthly average values of precipitation and 2-meter temperature two weeks in the future. These models are generally trained using the squared error loss function:

$$\ell_{\text{sq}-\text{err}}(\theta) = \mathbb{E}[(y - f_\theta(\mathbf{x}))^2]. \tag{2.1}$$

Tercile classification The goal of tercile classification is to predict whether the precipitation or 2-meter temperature will be "high" (above the 66th percentile, denoted $q = 1$), "medium" (between the 33rd and the 66th percentiles, denoted $q = 0$), or "low" (below the 33rd percentile, denoted $q = -1$). We compute these percentile values using the 1971-2000 climatology (see Section 2.2 for details), and these percentiles are computed for each calendar month m and location l pair. These models are generally trained using the cross-entropy loss function:

$$\ell_{\text{CE}}(\theta) = \mathbb{E}\left[\sum_{q=-1}^{1} -\mathbb{I}_{\{y=q\}} \log(f_\theta(\mathbf{x}))_q\right], \tag{2.2}$$

where $\mathbb{I}_{\{A\}} := \begin{cases} 1, & \text{if } A \text{ true} \\ 0, & \text{if } A \text{ false} \end{cases}$ is the indicator function and $(f_\theta(x))_q$ is the predicted probability that the target y corresponding to feature vector x will be in tercile q.

Quantile regression For a given percentile α, the goal of quantile regression is to predict the value z so that, conditioned on features x, the target y satisfies $y \le z$ with probability α. When we set α to a value close to one, such as $\alpha = 0.9$, this value z indicates what we can expect in "extreme outcomes", not just on average. These models are generally trained using the pinball loss function:

$$\ell_{\text{quantile}}(\theta) = \mathbb{E}[\rho_\alpha(y - f_\theta(\mathbf{x}))] \tag{2.3}$$

where

$$\rho_\alpha(z) := z(\alpha - \mathbb{I}_{\{(z<0)\}}) = \begin{cases} \alpha|z| & \text{if } z \ge 0 \\ (1-\alpha)|z| & \text{if } z < 0 \end{cases}. \tag{2.4}$$

2.4 Prediction methods

Our goal is to predict either the monthly average precipitation or the monthly average 2-meter temperature two weeks in advance (for example, we predict the average monthly precipitation for February on January 15). This section describes the notation used for features and targets, baselines and learning methods, and how spatial features are accounted for.

2.4.1 Notation

We let T denote the number of time steps used in our analysis, and L denote the number of spatial locations. We define the following variables:

- $u_{t,l}^{(k)}$ is the k-th ensemble member at time t and location l, where $k = 1, \ldots, K$, $t = 1, \ldots, T$, $l = 1, \ldots, L$. Every ensemble member represents the output of a given physics-based model forecast from different initial states.

- $v_{t,l}^{(p)}$ is the p-th observational variable, such as precipitation or temperature, geopotential height at 500mb, relative humidity near the surface, pressure at sea level and sea surface temperature, at time t and location l, with $p = 1, ..., P$.

- $\mathbf{z}_l^{(1)}, \mathbf{z}_l^{(2)}$ represent information about longitude and latitude of location l, respectively; each is a vector of length d. More details about this representation, called positional encoding (PE), can be found in Section 2.5.2.

- $\mathbf{x}_{t,l} := [u_{t,l}^{(1)}, ..., u_{t,l}^{(K)}, v_{t,l}^{(1)} ..., v_{t,l}^{(P)}, \mathbf{z}_l^{(1)}, \mathbf{z}_l^{(2)}]$ is a set of features at time t and location l.

- $y_{t,l}$ is the target – the ground truth monthly average precipitation or 2-meter temperature at the target forecast time $t + \delta_t$ at location l, where $\delta_t = 14$ days is our forecast horizon. For simplicity, we use a subscript t for $y_{t,l}$ instead of $t + \delta_t$ to match with the input features notation. The same holds for our ensuing definitions.

- $\hat{y}_{t,l}$ is the output of a forecast model for a given task at target forecast time $t + \delta_t$ and location l.

- $s_{m,l}$ – a 30-year mean (climatology) of an observed climate variable, such as precipitation or temperature, at a month $m = 1, \ldots, 12$ and location l.

- $\hat{s}_{m,l}$ – a 30-year climatology of a *predicted* climate variable, such as precipitation or temperature, at a month $m = 1, \ldots, 12$ and location l. For each location l and each month m, it is calculated as a mean of ensemble member predictions over the training period, as defined formally in Equation (2.7).

- $y_{t,l}^{\text{anomaly}}$ and $\hat{y}_{t,l}^{\text{anomaly}}$ are anomaly predictions and a true anomaly, at a month $m = 1, \ldots, 12$ and location l. They are used during evaluation. We define anomalies as

$$y_{t,l}^{\text{anomaly}} = y_{t,l} - s_{m(t),l}, \tag{2.5}$$

$$\hat{y}_{t,l}^{\text{anomaly}} = \hat{y}_{t,l} - s_{m(t),l}. \tag{2.6}$$

  For the special case of $\hat{y}$ corresponding to the ensemble mean, the ensemble members may exhibit bias, in which case we also consider $\hat{y}_{t,l}^{\text{anomaly}} = \hat{y}_{t,l} - \hat{s}_{m(t),l}$, where $\hat{s}_{m(t),l}$ is evaluated on the model's (ensemble mean) predictions:

$$\hat{s}_{m,l} := \frac{1}{T} \sum_{t=1}^{T} \hat{y}_{t,l} \mathbb{I}_{\{m=m(t)\}}, \; l = 1, \ldots, L. \tag{2.7}$$

  The model climatology $\hat{s}_{m,l}$ is computed using the training data. Note that we do not subtract the climatology from the input features and target variables, i.e., precipitation and temperature, when training our ML models. We subtract climatology from the model outputs only when evaluating their performance, as including climatology in the inputs to our ML models during training improves performance. Section 2.5.3 provides more details on the model evaluation.

In our analyses, the number of locations is $L = 3274$, there are $K = 24$ NCEP-CFSv2 ensemble members or $K = 11$ NASA-GMAO ensemble members. The ensemble members are used as input features to the learning-based methods as they are, we do not perform any feature extraction from them. The number of observational variables is usually $P = 17$. The details on these variables can be found in Section 2.2 and Section 2.5.1.

The target variable y is observed from 1985 to 2020. Data from January 1985 to September 2005 are used for training (249 time steps), and data from October 2005 to December 2010 are used for validation and model selection (63 time steps). Data from 2011 to 2020 (or from 2011 to 2018 in the case of NASA-GMAO data) are used to test our methods after all model development, selection, and parameter tuning are completed.

2.4.2 Baselines

Climatology It is the fundamental benchmark for weather and climate predictability. In particular, for a given time t, let $m(t) := (t \mod 12)$ correspond to the calendar month corresponding to t; then we compute the 30-year climatology of the target variable for a given location and time via

$$\hat{y}_{t,l}^{\text{hist}} = s_{m(t),l},\ t = 1, \dots, T,\ l = 1, \dots, L. \tag{2.8}$$

Ensemble mean This is the mean of all ensemble members for each location l at each time step t:

$$\hat{y}_{t,l}^{\text{ens mean}} := \frac{1}{K} \sum_{k=1}^{K} u_{t,l}^{(k)},\ t = 1, \dots, T,\ l = 1, \dots, L. \tag{2.9}$$

Linear regression Finally, we consider, as a baseline, a *linear regression* model applied to input features corresponding to ensemble member predictions: $\mathbf{x}_{t,l} = [u_{t,l}^{(1)}, \dots, u_{t,l}^{(K)}]$. Then, the model's output

$$\hat{y}_{t,l}^{\text{LR}} := \langle \boldsymbol{\theta}_l, \mathbf{x}_{t,l} \rangle + \theta_l^0, \tag{2.10}$$

where θ_l are the trained coefficients for input features for each location l, and θ_l^0 are the learned intercepts for each location l. Note that we train a different model for each spatial location, and the illustration for this model and its input's format is given in Figure 2.1(a).

2.4.3 Learning-based methods

Linear regression (LR) In contrast to the linear regression baseline, here other climate variables are added to the input features: $\mathbf{x}_{t,l} = [u_{t,l}^{(1)}, ..., u_{t,l}^{(K)}, v_{t,l}^{(1)}, ..., v_{t,l}^{(P)}]$. Then the model's output is defined with Equation (2.10). Because the feature vector is higher dimensional here than for the baseline, the learned θ_l is also higher dimensional. We train a different model for each spatial location. In our experiments with linear models, we do not include positional encoding $(\mathbf{z}_l^{(1)}, \mathbf{z}_l^{(2)})$ as input features, since they would be constants for each location's linear model.

In the context of regression, we minimize the squared error loss. The linear quantile regressor (Linear QR) is a linear model trained to minimize the quantile loss

$$\ell^{\mathrm{QR}} = \frac{1}{L}\sum_{l=1}^{L}\left[\frac{1}{T}\sum_{t=1}^{T}\rho_{\mathcal{T}}\left(y_{t,l} - \hat{y}_{t,l,}\right)\right], \tag{2.11}$$

where $\rho_{\mathcal{T}}$ is defined in Equation (2.4).

Random forest In the context of regression and tercile classification, we train a random forests that use ensemble predictions, the spatial location, and additional climate features to form the feature vector $\mathbf{x}_{t,l} = [u_{t,l}^{(1)}, ..., u_{t,l}^{(K)}, v_{t,l}^{(1)}, ..., v_{t,l}^{(P)}, \mathbf{z}_l^{(1)}, \mathbf{z}_l^{(2)}]$ for all location l and time t pairs. One random forest is trained to make predictions for any spatial location. The illustration for RF and its input's format is given in Figure 2.1(b): we train one RF model for all locations, and the spatial information is encoded as input features via PE vectors $\mathbf{z}_l^{(1)}, \mathbf{z}_l^{(2)}$.

In the context of quantile regression, we train a random forest quantile regressor (RFQR, Meinshausen [2006]), which grows trees the same way as the original random forest while storing all training samples. To make a prediction for a test point, the RFQR computes a weight for each training sample that corresponds to the number of leaves (across all trees in the forest) that contain the test sample and the training sample. The RFQR prediction is then a quantile of the weighted training samples across all leaves that contain the test sample. We show a figure representation of the RFQR in Section A.4.2. With this formulation, training a single RFQR for all locations is computationally demanding, so we train individual

RFQRs for every location.

Random forests are often referred as the best off-the-shelf classifiers [Hastie et al., 2009] even using the default hyperparameters [Biau and Scornet, 2016]. Our cross-validation (CV) and grid search experiments show that the RFs hyperparameters have little impact on the accuracy, and thus, we use the default parameters for RFs from the Scikit-learn library [Pedregosa et al., 2011].

Convolutional neural network To produce a forecast map for the U.S., we adapted a U-Net architecture [Ronneberger et al., 2015], which has an encoder-decoder structure with convolutional layer blocks. The U-Net maps a stack of images to an output image; in our context, we treat each spatial map of a climate variable or forecast as an image. Thus, the input to our U-Net is can be represented as a tensor composed of matrices: $\mathbf{X}_t = [\mathbf{U}_t^{(1)}, ..., \mathbf{U}_t^{(K)}, \mathbf{V}_t^{(1)}, ..., \mathbf{V}_t^{(P)}, \mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}]$.

Note that here, we use capital letters because the input to our U-Net consists of 2-D spatial maps, which are represented as matrices instead of vectors. The model output is a spatial map of the predicted target. This process is illustrated in Figure 2.1(c).

For the U-Net, we modify an available PyTorch implementation [Yakubovskiy, 2020]. The training set consists of 249 samples (images), which may be considered relatively limited for CNN training. To address this concern, we conduct bootstrapping experiments for the U-Net architecture, offering detailed insights into the impact of sample size on model performance. Further details are presented in Appendix A.3.2. We use a 10-fold CV over our training data and grid search to select parameters such as learning rate, weight decay, batch size, and number of epochs. The Adam optimizer [Kingma and Ba, 2014] is used in all experiments. After selecting hyperparameters, we train the U-Net model with those parameters on the full training dataset. The validation set is used to perform feature importance analysis. For regression, we train using squared error loss. In the context of quantile regression, we initialize the weights with those learned on squared error loss and then train on the quantile loss Equation (2.11).

Nonlinear model stacking Model stacking can improve model performance by combining the outputs of several models (usually called base models) [Pavlyshenko, 2018]. In our case, linear regression, random forests, and the U-Net are substantially different in architecture and computation, and we observe that they produce qualitatively different forecasts. We stack the linear model, random forest, and U-Net forecasts using a nonparametric approach:

$$\hat{y}_{t,l} = h(\hat{y}_{t,l}^{\text{LR}}, \hat{y}_{t,l}^{\text{RF}}, \hat{y}_{t,l}^{\text{UNET}}), \tag{2.12}$$

where h is a simple feed-forward neural network with a non-linear activation and $\hat{y}_{t,l}^{\text{LR}}, \hat{y}_{t,l}^{\text{RF}}, \hat{y}_{t,l}^{\text{UNET}}$ are the predictions of a linear model, random forest and the U-Net correspondingly and referred to as "base models". One stacking model is trained to make predictions for any spatial location. Figure 2.1(b) with input features that are predictions from other ML models and no PE vectors demonstrates the stacking model's framework. Model stacking can improve the forecast quality by combining predictions from three forecasting paradigms – spatial independence, conditional spatial independence, and spatial dependence (Section 2.4.4), and is analogous to the multi-model ensemble approach commonly used in weather and climate forecasting. The architecture details can be found in Appendix A.4.

We apply the following procedure for model stacking: the base models are first trained on half of the training data, and predicted values on the second half are used to train the stacking model h. Then, we retrain the base models on all the training data and apply the trained stacked model to the outputs of the base models. The proposed procedure helps to avoid overfitting.

2.4.4 Models of spatial variation

We consider three different forecasting paradigms. In the first, which we call the **spatial independence** model, we ignore all spatial information and train a separate model for each spatial location. In the second, which we call the **conditional spatial independence** model, we consider samples corresponding to different locations l as independent conditioned on the spatial location as represented by features $(\mathbf{z}_l^{(1)}, \mathbf{z}_l^{(2)})$. In this setting, a training sample corresponds to $(\mathbf{x}_i, y_i) = (\mathbf{x}_{t,l}, y_{t,l})$, where, with a small abuse of notation, we let i index a t, l pair. In this case, the number of training samples is $n = TL$. In the third paradigm, which we call the **spatial dependence** model, we consider a single training sample as corresponding to full spatial information (across all l) for a single t; that is $(\mathbf{X}_i, \mathbf{Y}_i) = ([x_{t,l}]_{l=1,\ldots,L}, [y_{t,l}]_{l=1,\ldots,L}])$, where now i indexes t alone. Models developed under the spatial dependence model account for the spatial variations in the features and targets. For instance, a convolutional neural network might input "heatmaps" representing the collection of physics-based model forecasts across the continental U.S. and output a forecast heatmap predicting spatial variations in temperature or precipitation instead of treating each spatial location as an independent sample.

Figure 2.1 shows general frameworks of these paradigms. All models combine information from all the different ensemble forecasts, and so in a broad sense, we can think of each prediction at a given time and location as a weighted sum of the ensemble forecasts across space, time, and ensemble members, where the weights are learned during the model training and may be data-dependent (i.e., nonlinear). From this perspective, we may think of different modeling paradigms as essentially placing different constraints on those weights:

- under spatial independence models, the weights may vary spatially but do not account for spatial correlations in the data;
- under conditional spatial independence models, the interpretation depends on the

model being trained – linear models have the same weights on ensemble predictions regardless of spatial location, while nonlinear models (e.g., random forests) have weights that may depend on the spatial location;

- under spatial dependence models, the weights vary spatially, depend on the spatial location, and account for spatial correlations among the ensemble forecasts and other climate variables.

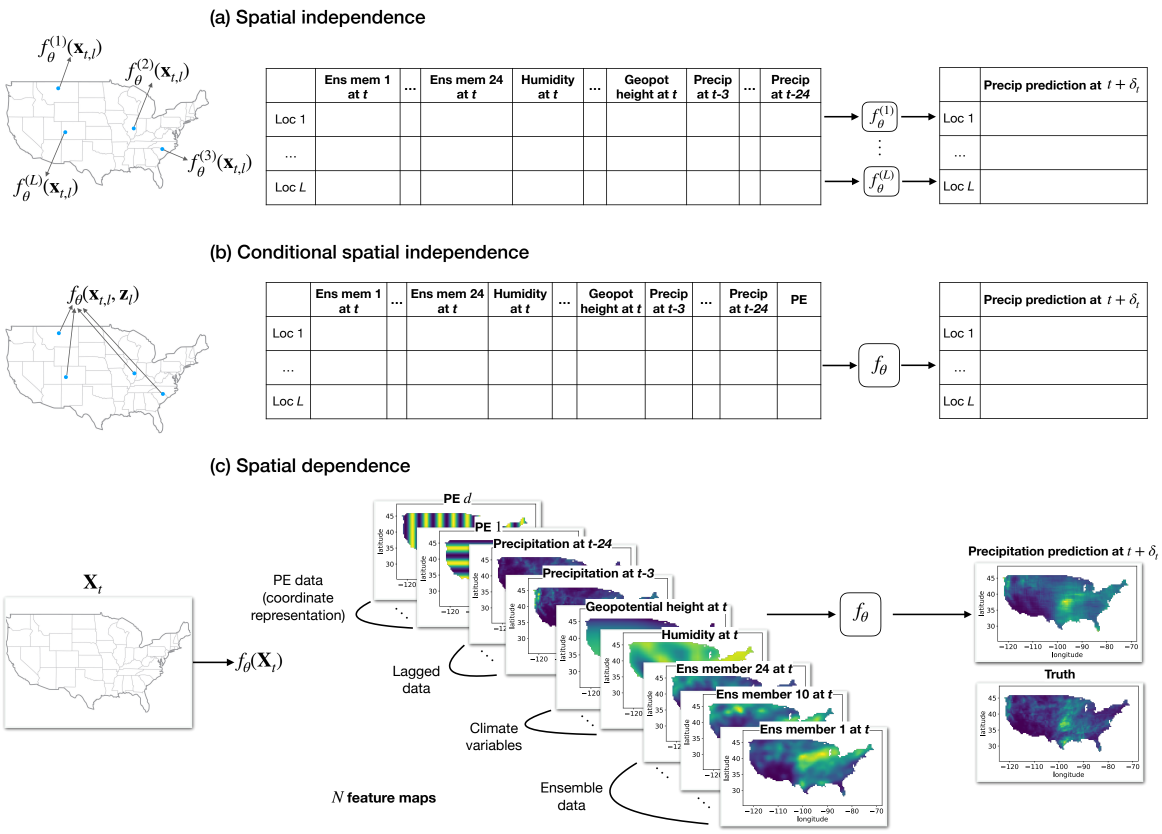


Figure 2.1: An illustration of different forecasting paradigms: (a) spatial independence models with a model for each spatial location, no accounting for spatial information; (b) conditional spatial independence models with one model for all locations, might consider the spatial information; (c) spatial dependence models that account for the spatial information by design. We replace “precipitation” in the illustration with “temperature” for temperature prediction, but the overall structure remains the same.

2.5 Experimental setup

This section provides details on the experimental setup, including positional encoding, removing climatology, and evaluation metrics. Data preprocessing details are presented in Appendix A.5.

2.5.1 Models' inputs details

Based on the available data, we use the following input features for our ML models:

- K ensemble forecasts for the target month,
- four climate variables: relative humidity, pressure, geopotential height, and temperature (if the target is precipitation) or precipitation (if the target is temperature) two months before the target month,
- the lagged target variable (the target variable two, three, four, twelve, and 24 months before the target date – five additional features),
- SSTs that are represented via principal components (PCs),
- and, finally, the positional embeddings.

SSTs are usually represented as eight PCs, and the embedding vector size is usually $d = 12$ as we describe Section 2.5.2. For example, using the NCEP-CFSv2 members, there are $\underbrace{24}_{K} + \underbrace{4+5+8}_{P} + \underbrace{12}_{d} \times 2 = 65$ input features for every time step and location. Figure 2.1 provides an illustration of these input features.

2.5.2 Positional encoding

Positional encoding [Vaswani et al., 2017] is a technique used in natural language processing (NLP) to inject positional information into data. In sequence-based tasks, such as language translation or text generation, the order of elements in the input sequence is important, but neural networks do not naturally capture this information. PE assigns unique encodings to each position in the sequence, which are then added to the original input before being processed by the model. This enables the model to consider the order and relative positions of elements, improving its ability to capture local and global context within a sequence and make accurate predictions [Devlin et al., 2018, Petroni et al., 2019, Narayanan et al., 2016]. This technique is helpful to represent the positional information outside the original NLP tasks [Gamboa, 2017, Gehring et al., 2017, Khan et al., 2022]. Several of our models use the spatial location as an input feature. Rather than directly using latitudes and longitudes, we use PE [Vaswani et al., 2017]:

$$\begin{aligned} z_l^{(1)}(i) =& \mathrm{PE}(l, 2i) = \sin(l/10000^{2i/d}), \\ z_l^{(2)}(i) =& \mathrm{PE}(l, 2i+1) = \cos(l/10000^{2i/d}), \end{aligned} \tag{2.13}$$

where l is a longitude or latitude value, $d = 12$ is the dimensionality of the positional encoding, and $i \in \{1, \dots, d\}$ is the index of the positional encoding vector. For the U-Net model, PE vectors are transformed into images in the following way: we take every value in the vector and fill the image of the desired size with this value. So, there are d images with the corresponding PE values. For the RF models, PE vectors can be used as they are.

2.5.3 Evaluation metrics

Regression metrics The forecast skill of our regression models is measured using the R^2 *value*. For each location l and ground-truth values $y_{t,l}$ and predictions $\hat{y}_{t,l}$ at this location, we compute

$$R_l^2 = 1 - \frac{\sum_{t=1}^{T}(y_{t,l}^{\text{anomaly}} - \hat{y}_{t,l}^{\text{anomaly}})^2}{\sum_{t=1}^{T}(y_{t,l}^{\text{anomaly}} - \bar{y}_l^{\text{anomaly}})^2}, \tag{2.14}$$

where

$$\bar{y}_l^{\text{anomaly}} = \frac{1}{T}\sum_{t=1}^{T} \hat{y}_{t,l}^{\text{anomaly}}.$$

Then, the average R^2 for all locations is calculated as

$$R^2 = \frac{1}{L}\sum_{l=1}^{L} R_l^2. \tag{2.15}$$

In addition to the average R^2 on the test data, we also estimate the median R^2 score across all U.S. locations.

We further report the *mean squared error* (MSE) of our predictions across all locations:

$$\text{MSE}_l := \frac{1}{T}\sum_{t=1}^{T} \left(y_{t,l} - \hat{y}_{t,l}\right)^2,$$

for $l = 1, \dots, L$, and

$$\text{MSE} = \frac{1}{L}\sum_{l=1}^{L} \text{MSE}_l. \tag{2.16}$$

We also report the standard error (SE), median, and 90th percentile of $\{\text{MSE}_l\}_l$. We say the difference between the two models is significant if their MSE $\pm$ SE intervals do not overlap. Note that the standard errors provided here should be used with caution since there are significant spatial correlations in the MSE values across locations, so we do not truly have L independent samples from an asymptotically normal distribution.

Tercile classification metrics We estimate the accuracy of our tercile classification predictions as the proportion of correctly classified samples out of all observations.

Quantile regression metrics For the quantile regression task, we report *mean quantile loss* from Equation (2.11) across all locations.

2.6 Experimental results

In this section, we report the predictive skill of different models applied to SSF over the continental U.S. using NCEP-CFSv2 ensemble members for regression and quantile regression. Precipitation forecasting is known to be more challenging compared to temperature forecasting [Knapp et al., 2011]. The results for the NASA-GMAO dataset are presented in Appendix A.1.1. The skill of different models on the tercile classification task is presented in Appendix A.2 for both datasets. Recall that all methods are trained on data spanning January 1985–September 2005, with data spanning October 2005 - December 2010 used for validation (i.e., model selection and hyperparameter tuning). Test data spanning 2011 to 2020 was **not** viewed at any point of the model development and training process and only used to evaluate the predictive skill of our trained models on previously unseen data; we refer to this period as the "test period". As a navigation tool for the reader, Table 2.2 gives references to the presented results for different tasks.

Table 2.2: A table with references to the main results.

Task	Data	Reference
Regression	precip NCEP-CFSv2	Table 2.3; Figure 2.2
	tmp NCEP-CFSv2	Table 2.4; Figure 2.3
	precip NASA-GMAO	Table A.1; Figure A.1
	tmp NASA-GMAO	Table A.2; Figure A.2
Quantile regression	precip NCEP-CFSv2	Table 2.5; Figure A.3
	tmp NCEP-CFSv2	Table 2.6; Figure 2.4
	precip NASA-GMAO	Table A.3; Figure A.4
	tmp NASA-GMAO	Table A.4; Figure A.5
Feature importance	precip NCEP-CFSv2	Table 2.10
	tmp NCEP-CFSv2	Table 2.11
Tercile classification	precip NCEP-CFSv2	Table A.5; Figure A.6
	tmp NCEP-CFSv2	Table A.6; Figure A.7
	precip NASA-GMAO	Table A.5; Figure A.7
	tmp NASA-GMAO	Table A.6; Figure A.9

2.6.1 Regression

Precipitation regression using NCEP-CFSv2 Precipitation regression results are presented in Table 2.3. While the individual ML approaches produce results generally similar to those of the baselines, the stacked ML model, in particular, outperforms the baseline models in almost all metrics. Note that the best R^2 value, associated with the stacked model, is still near zero; while this is a significant improvement over, for example, the ensemble mean, which has an R^2 value of -0.08, the low values for all methods indicate the difficulty of the forecasting problem. It is important to note that R^2 measures the accuracy of a model relative to a baseline corresponding to the mean of the target *over the test period* – that is, relative to a model that could never be used in practice as a forecaster because it uses future observations. The best *practical* analog to this would be the mean of the target *over the train period* – what we call the "historical mean" or climatology model. These two models are not the same, possibly because of the nonstationarity of the climate [Min et al., 2020]. Thus, even when our R^2 values are negative (i.e., we perform worse than the impractical mean of the target over the test period), we still perform much better than the practical climatology predictor. The model stacking approach is applied to the models trained on all available features (i.e., ensemble members, PE, climate variables; linear regression is trained on all features except PE). We decide what models to include in the stacking approach based on their performance on validation data. The low 90th percentile error implies that our methods not only have high skill on average but also that there are relatively few locations with large errors. While acknowledging the overall performance may not be exceptional, it is important to recognize the potential of machine learning methods in improving the quality of estimates relative to the standard baselines. To further evaluate the capabilities of the stacking approach, we also apply the approach to the baseline predictions, which include historical and ensemble means, as well as linear regression. The performance of the stacked baseline model exceeds that of any of the individual baseline models and is similar to the performance of the stacked ML approach in terms of the R^2 metric. However, the stacked ML approach outperforms it in all MSE-based metrics, indicating that the ML techniques can still provide additional skill even for as notoriously challenging a quantity as precipitation.

Table 2.3: Results for precipitation regression the using NCEP-CFSv2 ensemble, with errors reported over the test period. LR refers to linear regression on all features, including ensemble members, lagged data, climate variables, and SSTs. ML model stacking is performed on models that are trained on all features. The **best** results are in bold. MSE is reported in squared mm.

Model	Features	Mean R^2 (↑)	Median R^2 (↑)	Mean Sq Err (↓)	Median MSE (↓)	90th prctl MSE (↓)
Baseline	Climatology	-0.06	-0.01	2.33 ± 0.04	1.59	4.96
	Ens mean	-0.08	0.01	2.19 ± 0.04	1.55	4.57
	Linear Regr	-0.11	-0.07	2.26 ± 0.04	1.54	4.72
	Baseline stacking	0.00	0.04	2.15 ± 0.04	1.44	4.55
LR	All features	-0.33	-0.25	2.71 ± 0.05	1.91	5.45
U-Net	All features	-0.10	-0.01	2.18 ± 0.03	1.44	4.62
RF	All features	-0.11	-0.01	2.17 ± 0.05	1.48	4.45
Stacked	LR, U-Net, RF outputs	**0.02**	**0.04**	**2.07 ± 0.03**	**1.42**	**4.38**

Figure 2.2 illustrates performance of key methods with R^2 heatmaps over the U.S. to highlight spatial variation in errors. The RF and U-Net R^2 fields are qualitatively similar, but they are still quite different in certain states such as Georgia, North Carolina, Virginia, Utah, and Colorado. The LR map is noticeably poor across most of the regions. The stacked ML model's heatmap reveals large regions where its predictive skill exceeds that of all other methods. Note that model stacking yields relatively accurate predictions even in regions where the three constituent models individually perform poorly (e.g., southwestern Arizona), highlighting the generalization abilities of our stacking approach. All methods tend to have higher accuracy on the Pacific Coast, in the Midwest, and in southern states such as Alabama and Missouri. The stacking model heatmaps both look similar. The stacking model applied to the baselines has better R^2 scores in California compared to the stacked ML methods. However, the stacked ML model reveals larger positive R^2 regions and fewer dark red spots, particularly evident in New Mexico, Minnesota, and Utah.

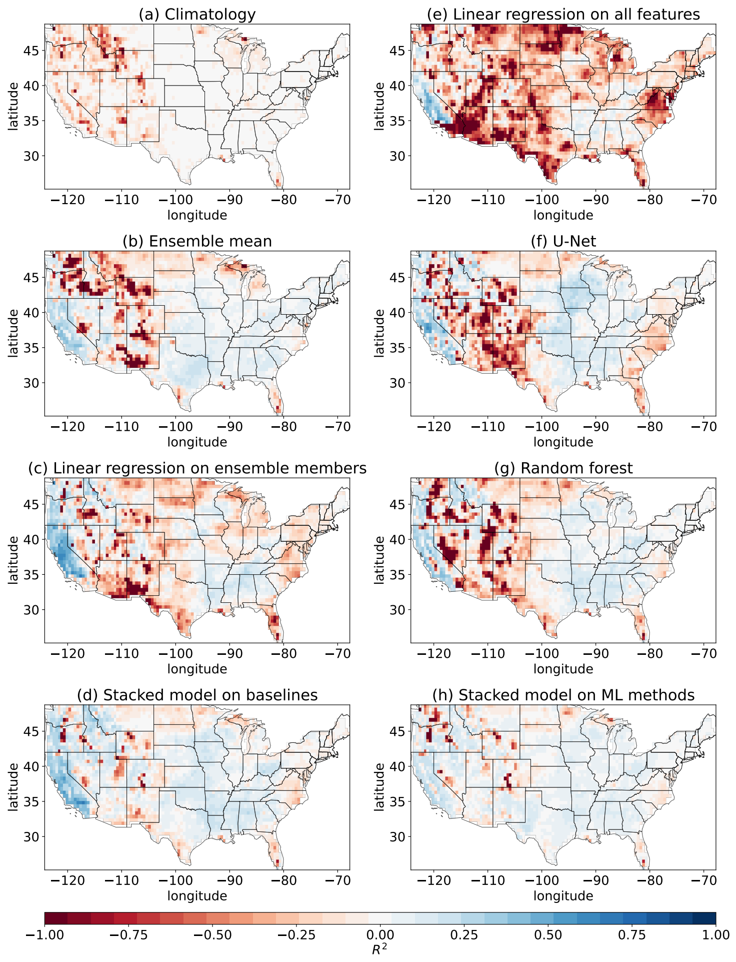


Figure 2.2: R^2 score heatmaps of baselines and learning-based methods for precipitation regression using NCEP-CFSv2 ensemble members; errors are computed over the test period. Positive values (blue) indicate better performance. See Section 2.6.1 for details.

Temperature regression Table 2.4 shows results for 2-meter temperature regression. The learning-based models, especially the random forest and stacked model, significantly outperform the baseline models in terms of MSE and R^2 score. The random forest also outperforms linear regression and the U-Net. Note that LR, U-Net, and RF are trained without using SST information since SST features yielded worse performance over the validation period.

Figure 2.3 illustrates the performance of these methods with R^2 heatmaps over the U.S. As expected, the model stacking approach shows the best results across spatial locations. We notice that there are still regions such as the West, some regions in Texas, Florida, and

Table 2.4: Results for temperature regression using the NCEP-CFSv2 ensemble, with errors reported over the test period. LR refers to linear regression on all features including ensemble members, lagged data, land variables. Model stacking is performed on models that are trained on all features except SSTs. The **best** results are in bold. MSE is reported in squared $^{\circ}C$.

Model	Features	Mean R^2 (↑)	Median R^2 (↑)	Mean Sq Err (↓)	Median MSE (↓)	90th prctl MSE (↓)
Baseline	Climatology	-0.66	-0.17	6.57 ± 0.11	5.04	9.99
	Ens mean	-0.47	0.08	5.51 ± 0.10	3.83	9.16
	Linear Regr	0.04	0.17	3.60 ± 0.03	3.25	5.49
LR	All features w/o SSTs	0.05	0.16	3.57 ± 0.02	3.33	5.41
U-Net	All features w/o SSTs	0.01	0.18	3.65 ± 0.02	3.38	5.31
RF	All features w/o SSTs	0.16	0.25	3.17 ± 0.02	2.99	4.63
Stacked	LR, U-Net, RF outputs	**0.18**	**0.27**	**3.11 ± 0.02**	**2.93**	**4.56**

Georgia where all models tend to achieve a negative R^2 score.

2.6.2 Quantile regression

We explore the use of quantile regression to predict values z so that "there's a 90% chance that the average temperature will be below z° at your location next month" – or, equivalently, "there is a 10% chance that the average temperature will exceed z° at your location next month." In this sense, quantile regression focused on the 90-th percentile predicts temperature and precipitation extremes, a task highly relevant to many stakeholders. We train a linear regression model fitting the quantile loss (Linear QR), a random forest quantile regressor (RFQR), [Meinshausen, 2006], a U-Net, and the stacked model. The Linear QR and the RFQR details are discussed in Section 2.4. The below experimental results show that temperature extremes can be predicted with high accuracy by the learning-based models (particularly our stacked model), in stark contrast to historical quantiles or ensemble quantiles in the case of temperature quantile regression. The results for precipitation are less striking overall, though the learned models are significantly more predictive in some locations on this quantile regression task.

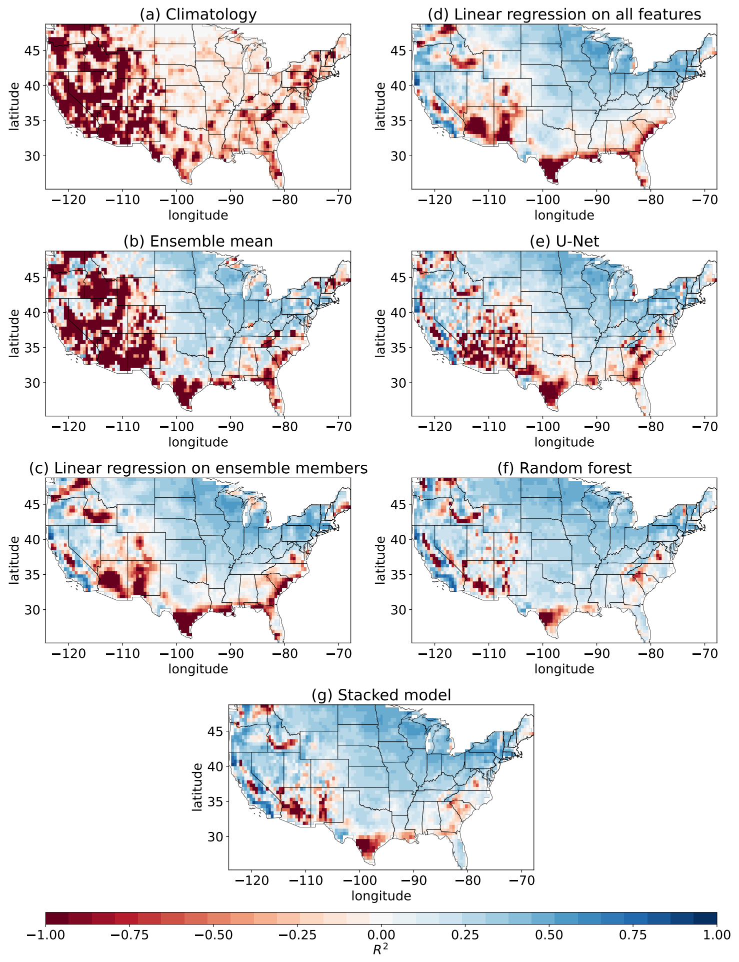


Figure 2.3: R^2 score heatmaps of baselines and learning-based methods for temperature regression using NCEP-CFSv2 ensemble members; errors are computed over the test period. Positive values (blue) indicate better performance. See Section 2.6.1 for details.

Quantile regression of precipitation For each location, the 90th percentile value is calculated based on the historical data. For the ensemble 90th percentile, we simply take the 90th percentile of the K ensemble members. Table 2.5 summarizes results for precipitation quantile regression using the NCEP-CFSv2 ensemble. Our stacked model is able to significantly outperform all baselines. The performance illustration is given in Appendix A.1, Figure A.3.

Quantile regression of temperature Table 2.6 summarizes results for temperature quantile regression using the NCEP-CFSv2 ensemble. Note that we do not include SST

Table 2.5: Test results for precipitation quantile regression using NCEP-CFSv2 dataset, with target quantile = 0.9. Linear QR refers to a linear quantile regressor. RFQR corresponds to a Random Forest Quantile Regressor. Model stacking is performed on models that are trained on all features. Our ML methods are able to significantly outperform all baselines. The **best** results are in bold. Quantile loss is reported in mm.

Model	Features	Mean Qtr Loss (↓)	Median Qtr Loss (↓)	90th prctl Qtr Loss (↓)
Baseline	Historical 90th percentile	0.304 ± 0.003	0.278	0.504
	Ens 90th percentile	0.311 ± 0.003	0.275	0.488
	Linear QR ens only	0.310 ± 0.003	0.266	0.505
Linear QR	All features	0.287 ± 0.003	0.248	0.463
U-Net	All features	0.312 ± 0.002	0.281	0.504
RFQR	All features	**0.282** ± **0.002**	0.257	**0.453**
Stacked	U-Net, RFQR, LQR outputs	**0.282** ± **0.002**	**0.256**	0.457

features for temperature quantile regression in our learned models. We observe that all of our learned models are able to significantly outperform all baselines. In Figure 2.4, we show the heatmaps of quantile loss of baselines and our learned models. We observe that the learned models produce predictions with varied quality, and the stacked model can pick up useful information from them. For example, in Arizona and Texas, the Linear QR, U-Net, and RFQR show some errors but in different locations, and the stacked model can exploit the advantages of each model.

Table 2.6: Test results for temperature quantile regression using NCEP-CFSv2 dataset, with target quantile = 0.9. Linear QR refers to a linear quantile regressor. RFQR corresponds to a Random Forest Quantile Regressor. Model stacking is performed on models that are trained on all features except for SSTs. Learned models can predict highly likely temperature ranges accurately, meaning there are fewer unpredicted temperature spikes. The **best** results are in bold. Quantile loss is reported in $^{\circ}C$.

Model	Features	Mean Qtr Loss (↓)	Median Qtr Loss (↓)	90th prctl Qtr Loss (↓)
Baseline	Historical 90th percentile	0.589 ± 0.008	0.435	0.980
	Ens 90th percentile	0.642 ± 0.009	0.468	1.196
	Linear QR ens only	0.336 ± 0.004	0.286	0.488
Linear QR	All features w/o SSTs	0.318 ± 0.002	0.301	0.407
U-Net	All features w/o SSTs	0.363 ± 0.003	0.329	0.488
RFQR	All features w/o SSTs	0.320 ± 0.002	0.307	0.384
Stacked	U-Net, RFQR, LQR outputs	**0.287** ± **0.001**	**0.285**	**0.344**

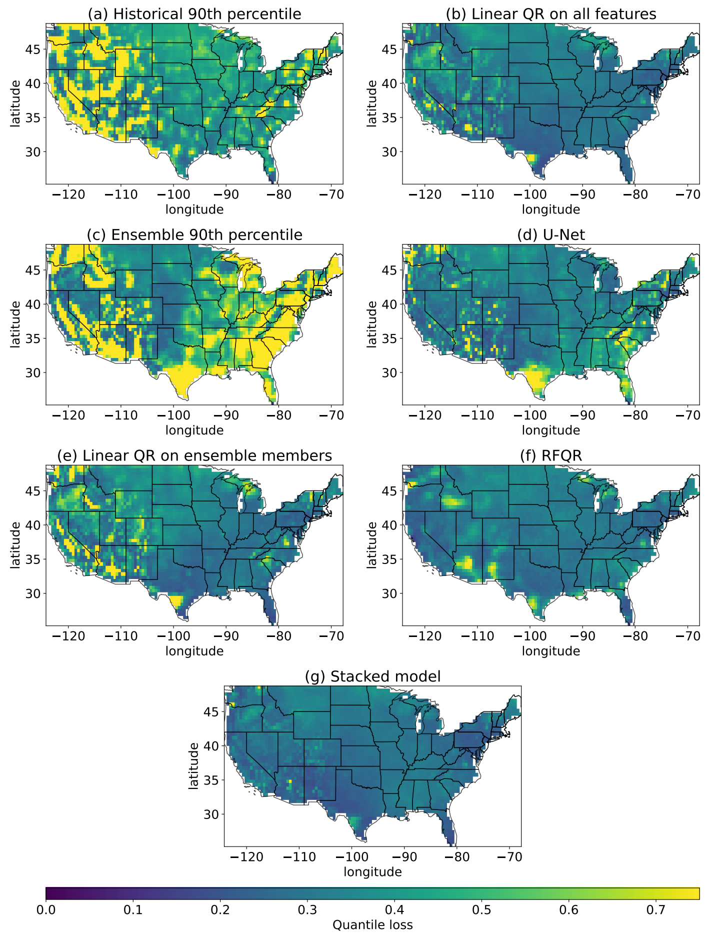

(a) Historical 90th percentile
(b) Linear QR on all features
(c) Ensemble 90th percentile
(d) U-Net
(e) Linear QR on ensemble members
(f) RFQR
(g) Stacked model
latitude
longitude
45
40
35
30
−120
−110
−100
−90
−80
−70
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Quantile loss

Figure 2.4: Test quantile loss heatmaps of baselines and learning-based methods for temperature quantile regression using NCEP-CFSv2 dataset. Blue regions indicate smaller quantile loss. See Section 2.6.2 for details.

2.7 Discussion

2.7.1 The efficacy of machine learning for SSF

Several hypotheses might explain why ML may be a promising approach for SSF, and we probe those hypotheses in this section.

Using full ensemble vs. ensemble mean Past works use ensemble mean as an input feature to machine learning methods in addition to the climate variables [Hwang et al., 2018, He et al., 2021]. Ensembles provide valuable information not only about expected climate behavior but also variance or uncertainty in multiple dimensions; methods that rely solely on ensemble mean lack information about this variance. Ensemble members may have systematic errors, either in the mean or the variability, arising from different initial conditions of the corresponding dynamic model that are not readily apparent to users. The more recently initialized an ensemble member is, the better it usually performs. While taking the average of these ensemble members may cancel out the deficiencies of each individual member, it is also possible that details of each member's systematic errors may be directly discovered and corrected independently by a machine learning model. Therefore, using a single ensemble statistic, such as the ensemble mean, as a feature may not fully capitalize on the information provided by using all members of the lagged ensemble as features.

In our experiments, we find that using all available ensemble members enhances the prediction quality of our approaches. As an illustration, we show the results of the LR, RF, U-Net, and the stacked model trained on all ensemble members, compared to the ML models trained on the ensemble mean only. In addition to the full ensemble or the ensemble mean, we use other available features (as in our previous regression results). Table 2.7 and Table 2.8 demonstrate the precipitation and temperature forecasting results. For the linear regression, utilizing the ensemble mean with all other features produces the best test performance compared to using the full ensemble. Such behavior is not surprising for the LR since the full ensemble incorporates large variance across ensemble members, which may result in a worse linear fit. For the U-Net, RF, and stacked model, we observe significant performance improvements, in terms of having at least one standard error smaller MSE, when using the full ensemble instead of using the ensemble mean. When we compare the performance of learned models using only the ensemble mean to that of the learned models that use both the ensemble mean and the ensemble standard deviation for each spatial location, we find that the addition of the standard deviation feature does not provide enough information to significantly improve the performance of ML models, and in fact the

U-Net that exhibits a performance degradation – a potential a sign of overfitting. These observations are visually supported by Figure 2.5 and Figure 2.6, where the R^2 heatmaps of our methods (except the U-Net) utilizing ensemble mean and standard deviation closely resemble the performance of methods solely relying on ensemble mean. We conclude that the full ensemble contains important information for SSF aside from the ensemble mean, and our models can capitalize on this information for precipitation and temperature forecasting.

We can perform a statistical test to verify that the performance discrepancies between using the ensemble mean and using the full ensemble are statistically significant for the stacked model. As before, let $\hat{y}_{t,l}$ refer to the estimate under our usual stacked model (i.e., with all ensemble members). Let $\hat{y}_{t,l}^{\text{SEA}}$ refer to a stacked model with just the ensemble mean as a feature, instead of all ensemble members. We can employ a sign test framework [DelSole and Tippett, 2014, Cash et al., 2019] to compare model performance under minimal distributional assumptions. Namely, we only make the following i.i.d. assumption over the time dimension:

$$|\hat{y}_{t,l} - y_{t,l}| < |\hat{y}_{t,l}^{\text{SEA}} - y_{t,l}| \overset{\text{iid}}{\sim} \text{Bernoulli}(p_l)$$

Intuitively, this corresponds to assuming it is a coin flip which model will perform better at each time point and location, and we would like to test whether each location's "coin" is fair or not. We can then formulate our null and alternative hypotheses for each location l as follows:

$$H_{0,l} : p_l = 0.5, \quad H_{0,l} : p_l > 0.5$$

Thus, our overall test for significance is for the global null hypothesis $H_0 = \cap_{l=1}^{3274} H_{0,l}$. We calculate a p-value for each $H_{0,l}$, and then we check whether any of these p-values is below a Bonferroni-corrected threshold of $0.05/3274 = 1.53 \times 10^{-5}$, where 3274 refers to the number of locations. In fact, the minimum p-values for this test with precipitation and regression alike are far below this threshold (1.68×10^{-10} and 4.42×10^{-24}, respectively). This allows us to reject the global null hypothesis for both temperature and precipitation, and we conclude that including the full ensemble in our stacked model significantly outperforms including just the ensemble mean.

Table 2.7: Precipitation forecasting performance comparison of the LR, RF, U-Net, and stacked model trained using the ensemble mean, using the sorted ensemble members, or using the original ensemble, in addition to other features. Scores on the test data are reported, and NCEP-CFSv2 data is used. The **best** results are in bold. MSE is reported in squared mm.

Model	Features	Mean R^2 (↑)	Mean Sq Err (↓)
LR	Ensemble mean + all features	**-0.28**	**2.59**±0.04
	Ensemble mean & std + all features	-0.29	2.61 ± 0.04
	Shuffled ensemble + all features	-0.41	2.84±0.05
	Sorted ensemble + all features	-0.43	2.87±0.05
	Full ensemble + all features	-0.33	2.71±0.05
U-Net	Ensemble mean + all features	-0.45	2.76±0.04
	Ensemble mean & std + all features	-0.25	2.65 ± 0.04
	Shuffled ensemble + all features	-0.27	2.77±0.05
	Sorted ensemble + all features	-0.43	2.78±0.04
	Full ensemble + all features	**-0.1**	**2.18**±0.03
RF	Ensemble mean + all features	-0.16	2.36±0.04
	Ensemble mean & std + all features	-0.15	2.32 ± 0.04
	Shuffled ensemble + all features	-0.16	2.29±0.04
	Sorted ensemble + all features	-0.18	2.30±0.04
	Full ensemble + all features	**-0.11**	**2.17**±0.05
Stacked	Ensemble mean + all features	-0.08	2.28±0.04
	Ensemble mean & std + all features	-0.05	2.26 ± 0.04
	Shuffled ensemble + all features	-0.04	2.25±0.04
	Sorted ensemble + all features	-0.11	2.24±0.04
	Full ensemble + all features	**0.02**	**2.07**±0.03

Sensitivity to ensemble formulation We consider the hypothesis that there is a set of k ensemble members that are always best. To test this hypothesis, we use a training period to identify which k members perform best for each location, and then during the test period, compute the average of only these k ensemble members. The performance of this approach depends on k, the number of ensemble members we allow to be designated "good." We have not found that the performance for any k exhibits a significant improvement over

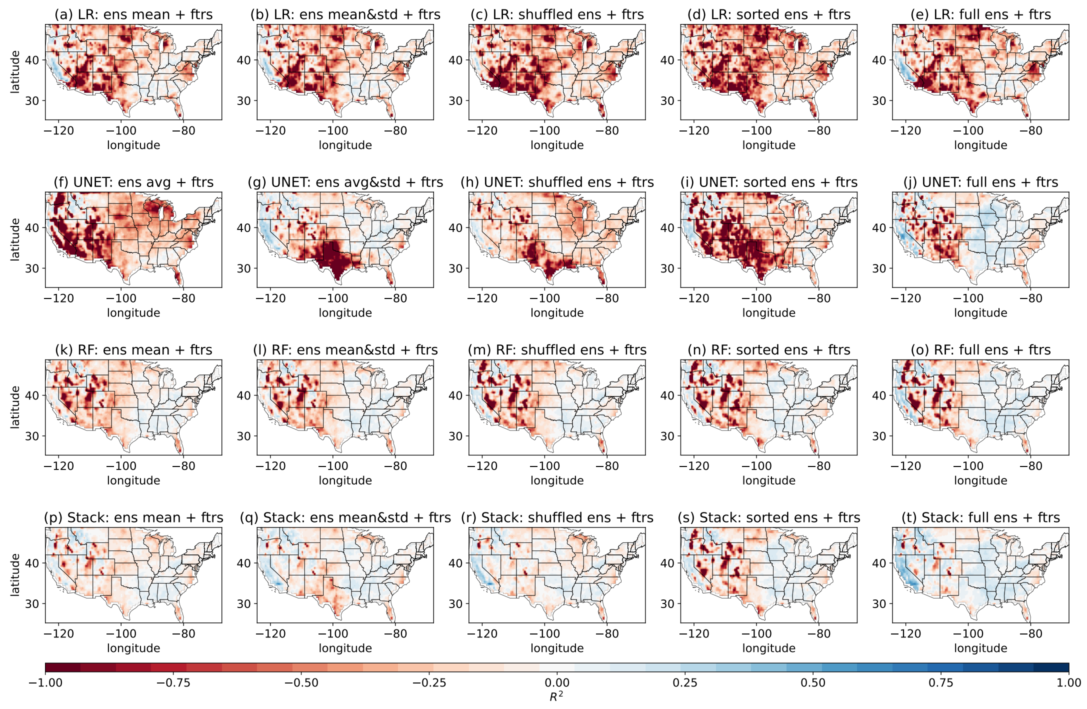

(a) LR: ens mean + ftrs
(b) LR: ens mean&std + ftrs
(c) LR: shuffled ens + ftrs
(d) LR: sorted ens + ftrs
(e) LR: full ens + ftrs
(f) UNET: ens avg + ftrs
(g) UNET: ens avg&std + ftrs
(h) UNET: shuffled ens + ftrs
(i) UNET: sorted ens + ftrs
(j) UNET: full ens + ftrs
(k) RF: ens mean + ftrs
(l) RF: ens mean&std + ftrs
(m) RF: shuffled ens + ftrs
(n) RF: sorted ens + ftrs
(o) RF: full ens + ftrs
(p) Stack: ens mean + ftrs
(q) Stack: ens mean&std + ftrs
(r) Stack: shuffled ens + ftrs
(s) Stack: sorted ens + ftrs
(t) Stack: full ens + ftrs
latitude
longitude
R^2

Figure 2.5: Precipitation regression test R^2 heatmaps of LR, U-Net, RF, and stacked model trained using ensemble mean only, using sorted and shuffled ensemble, or using the full ensemble. The NCEP-CFSv2 ensemble is used. See Section 2.7.1 for details.

the ensemble mean.

If the ensemble members have different levels of accuracy over various seasons, locations, and conditions, then a machine learning model may be learning when to “trust" each member. We know that our ensemble members are lagged, meaning they are initialized at different times. We believe each ensemble member encapsulates valuable information derived from the underlying physical model during each initialization. To investigate the impact of ensemble member order, we perform the following experiment: we randomly permute ensemble members at every time step t for all locations (preserving the spatial information) and apply our ML models to these shuffled ensembles. From Table 2.7 and Table 2.8, this approach negatively affects the performance of the ML models compared to using the full ensemble with the original order. One possible explanation is that the learned models lose the ability to learn which ensemble member to trust, as this information is tied to the initialization

Table 2.8: Temperature forecasting performance comparison of the LR, RF, U-Net, and stacked model trained using the ensemble mean, using the sorted ensemble members, or using the original ensemble, in addition to other features. Scores on the test data are reported and NCEP-CFSv2 data is used. The **best** results are in bold. MSE is reported in squared $^{\circ}C$.

Model	Features	Mean R^2 ($\uparrow$)	Mean Sq Err ($\downarrow$)
LR	Ensemble mean + all features	**0.06**	**3.55**±0.03
	Ensemble mean & std + all features	0.05	3.59±0.03
	Shuffled ensemble + all features	0.03	3.95±0.03
	Sorted ensemble + all features	-0.02	3.87±0.03
	Full ensemble + all features	0.05	3.57±0.02
U-Net	Ensemble mean + all features	0.00	3.77±0.03
	Ensemble mean & std + all features	0.19	4.61±0.03
	Shuffled ensemble + all features	-0.29	4.75±0.03
	Sorted ensemble + all features	-0.94	6.51±0.05
	Full ensemble + all features	**0.01**	**3.65**±0.02
RF	Ensemble mean + all features	0.10	3.57±0.02
	Ensemble mean & std + all features	0.10	3.56 ±0.02
	Shuffled ensemble + all features	0.05	3.65±0.02
	Sorted ensemble + all features	0.10	3.44±0.02
	Full ensemble + all features	**0.16**	**3.17**±0.02
Stacked	Ensemble mean + all features	0.11	3.43±0.02
	Ensemble mean & std + all features	0.13	3.30 ±0.02
	Shuffled ensemble + all features	0.08	3.47±0.02
	Sorted ensemble + all features	0.03	3.70±0.02
	Full ensemble + all features	**0.18**	**3.11**±0.02

time of each ensemble member. Even though the spatial information remains intact after the shuffling, the models can no longer exploit dependencies associated with the original ensemble structure.

Additionally, we conduct an experiment designed to test whether it is important to keep track of which ensemble member made each prediction or whether it is the *distribution* of predictions that is important. The modeling approach for the former would be to feed in ensemble member 1's forecast as the first feature, ensemble member 2's forecast as the second feature, etc. The modeling approach under the distributional hypothesis is to make the smallest prediction be the first feature, the second-smallest prediction be the second feature, and so on – i.e., we sort the ensemble forecasts for each location separately. Note

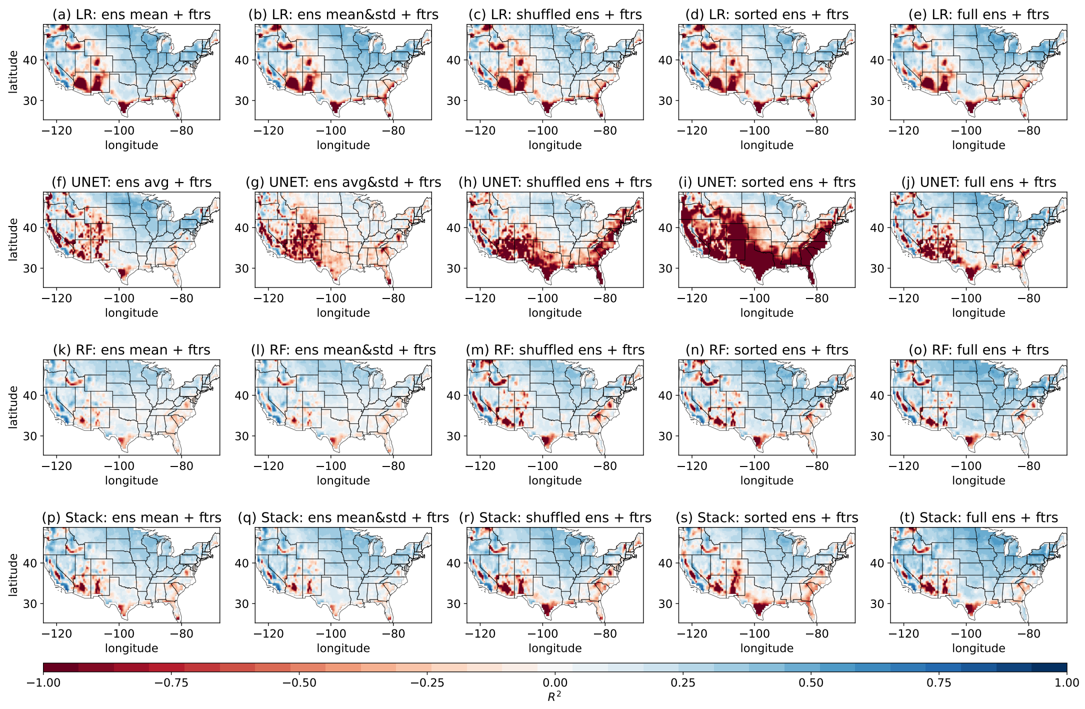

(a) LR: ens mean + ftrs
(b) LR: ens mean&std + ftrs
(c) LR: shuffled ens + ftrs
(d) LR: sorted ens + ftrs
(e) LR: full ens + ftrs
(f) UNET: ens avg + ftrs
(g) UNET: ens avg&std + ftrs
(h) UNET: shuffled ens + ftrs
(i) UNET: sorted ens + ftrs
(j) UNET: full ens + ftrs
(k) RF: ens mean + ftrs
(l) RF: ens mean&std + ftrs
(m) RF: shuffled ens + ftrs
(n) RF: sorted ens + ftrs
(o) RF: full ens + ftrs
(p) Stack: ens mean + ftrs
(q) Stack: ens mean&std + ftrs
(r) Stack: shuffled ens + ftrs
(s) Stack: sorted ens + ftrs
(t) Stack: full ens + ftrs
latitude
longitude
40
30
−120
−100
−80
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
R^2

Figure 2.6: Temperature regression test R^2 heatmaps of LR, U-Net, RF and stacked model trained using ensemble mean only, using sorted and shuffled ensemble, or using the full ensemble. The NCEP-CFSv2 ensemble is used. See Section 2.7.1 for details.

that this entails treating the ensemble members symmetrically: the model would give the same prediction if ensemble member 1 predicted a and ensemble member 2 predicted b or if ensemble member 1 predicted b and ensemble member 2 predicted a. In statistical parlance, this is passing in the order statistics of the forecasts as the features rather than their original ordering. Note that for NCEP-CFSv2, ensemble forecasts are originally ordered according to the time their initial conditions are set [Saha et al., 2014]. According to Table 2.7 and Table 2.8, using the sorted ensemble drastically degrades U-Net's performance, which is essentially because we sort the ensemble members for each location individually, and sorting the ensemble members individually for each location may hamper the ability for the U-Net to learn spatial structure. In the case of precipitation regression with the stacking model from Table 2.7, the MSE of the sorted approach is 2.24, which is worse than the 2.07 MSE for using the original ordering. In the case of temperature forecasting from Table 2.8, the

MSE of the sorted approach is 3.70, which is much worse than the 3.11 MSE for using the original ordering. The mean R^2 of the sorted approach is also lower compared to the original ordering. In both cases, the performance is better when we feed in the features in such a way that the machine learning model has an opportunity to learn aspects of each ensemble member, not merely their order statistics. Therefore, imposing a symmetric treatment of ensemble members degrades performance. Figure 2.5 and Figure 2.6 shows the corresponding R^2 heatmaps of our models for precipitation and temperature regression tasks.

2.7.2 Using spatial data

There are several ways to incorporate information about location in our models. U-Net has access to spatial dependencies through its design. Specifically, our U-Net inputs the spatial location of each point in the map. Naively, we might represent each location using the latitude and longitude values. Alternatively, we may use positional encoding, which is known to be beneficial in many ML areas, not only in NLP (as we mention in Section 2.5.2). PE captures the order (or position) and allows one to learn the contextual relationships (local context – relationships between nearby elements and global context dependencies across the entire sequence). We assume that the PE approach represents spatial information in a manner more accessible to our learned models.

As an illustration, Table 2.9 and Figure 2.7 demonstrate the performance of a stacked model using LR, RF, and U-Net trained using positional encoding, using latitude and longitude values and using no features representing the spatial information (no PE and no latitude or longitude values). Other inputs to the LR, RF, and U-Net models are ensemble member forecasts, lagged target variable, climate variables, and SSTs (except in the case of temperature forecasting). The results suggest that using PE enhances the predictive skill of our models, compared to using just the lat/lon values or no location information, especially for the temperature forecasting task. Using no information about locations hurts the performance of precipitation regression. Thus, our models can account for spatial dependencies using input features, and PE is more beneficial than the raw latitude and longitude information. These findings on PE effectiveness are consistent with prior findings in ML. For example, [Wu et al., 2021] investigate the efficacy of PE in the context of the visual transformer model used for image classification and object detection. We show a more detailed analysis with results for the LR, RF, and U-Net in Appendix A.3.1.

Table 2.9: Test performance comparison of the stacked model of LR, RF, and U-Net trained using no spatial features, using latitude and longitude values, or using PE. Utilizing spatial representations, including PE, latitude, and longitude values, helps advance the predictive skill. Furthermore, using positional encoding is more beneficial than using raw latitude and longitude values. The **best** results are in bold. MSE is reported in squared mm for precipitation and in squared $^\circ C$ for temperature.

Target	Features	Mean R^2 (↑)	Mean Sq Err (↓)
Precip	All + no location info	-0.05	2.13±0.03
	All + lat/lon values	-0.01	2.21±0.04
	All + PE	**0.02**	**2.07**±0.03
Tmp	All + no location info	0.12	3.35±0.02
	All + lat/lon values	0.12	3.33±0.02
	All + PE	**0.18**	**3.11**± 0.02

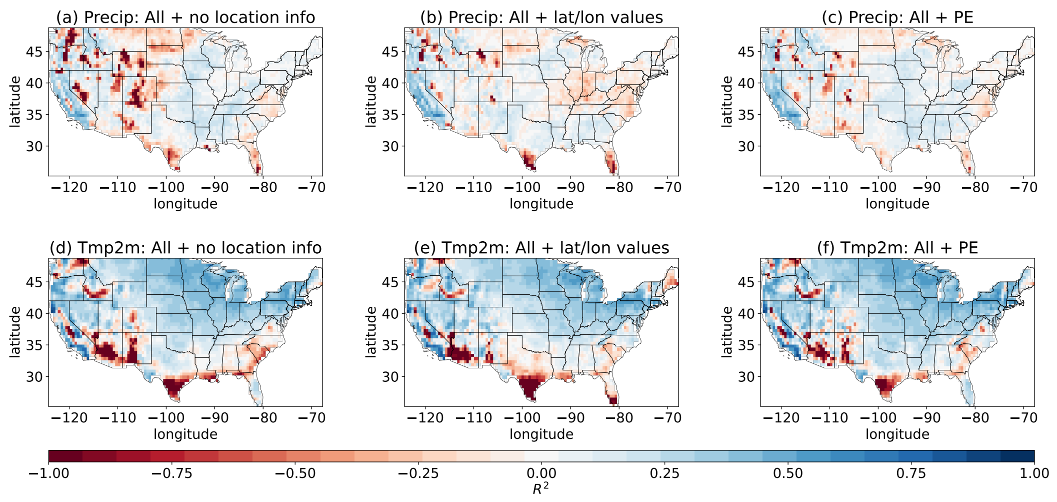


Figure 2.7: Test R^2 heatmaps of the stacked model of LR, RF, and U-Net trained using no spatial features, using latitude and longitude values or using PE. The NCEP-CFSv2 ensemble is used. See section 2.7.2 for details.

2.7.3 Variable importance

One consideration when implementing ML for SSF is that ML models can incorporate side information (such as spatial information, lagged temperature and precipitation values, and climate variables). In this section, we explore the importance of the various components of side information. We see that including the observational climate variables improves the performance of the random forest and the U-Net when doing precipitation regression. Furthermore, including positional encoding of the locations improves the performance of the U-Net, while the principle components of the sea surface temperature do not make a notable difference in the case of temperature prediction.

More specifically, Table 2.10 summarizes grouped feature importance of precipitation regression using the NCEP-CFSv2 ensemble. We observe that models, in particular random forest and U-Net, trained on all available data achieve the best performance. In the case of linear regression, the SSTs features are neither very helpful nor actively harmful. Therefore, to be consistent, we use predictions of these models trained on all features as input to the stacking model.

Table 2.11 summarizes grouped feature importance of temperature regression using the NCEP-CFSv2 ensemble. In this case, adding some types of side information may yield only very small improvements to predictive skill, and in some cases, the additional information may decrease predictive skill. On the one hand, this effect can be explained by different training set sizes for different models: as we outline in Section 2.4.4, the training set size for RF is $n = TL$, while $n = T$ for U-Net. This effect also may be a sign of overfitting, as temperature forecasting presents a comparatively less complex challenge than precipitation forecasting. We also note that SSTs provide only marginal (if any) improvement in predictive skill, in part because Pacific SSTs are less helpful away from the western U.S. [Mamalakis et al., 2018, Seager et al., 2007]. It could also be that information from the SSTs is already being well-captured by the output from the dynamical models, and thus, including observed

Table 2.10: Grouped feature importance results on validation for precipitation regression task using NCEP-CFSv2 ensemble members. The results suggest that using additional observational information helps to improve the performance of learning-based models for this task. –"– means a repetition of features that are used above. For example, in the U-Net part of the table, "–"– & lags" means that ensemble members, PE, and lags are used as features and "–"– & SSTs" means ensemble members, PE and lags, land features and SSTs are used as features. The **best** results are in bold. MSE is reported in squared mm.

Model	Features	Mean R^2 (↑)	Median R^2 (↑)	Mean Sq Err (↓)	Median MSE (↓)	90th prctl MSE (↓)
LR	Ens members	-0.13	-0.08	2.11 ± 0.03	1.53	4.63
	–"– & lags	-0.11	-0.07	2.10 ± 0.03	1.50	4.59
	–"– & climate variables (no SSTs)	**-0.09**	**-0.06**	**2.06** ± 0.03	**1.47**	**4.52**
	–"– & SSTs	-0.10	-0.07	2.08 ± 0.03	1.47	4.61
U-Net	Ens members with PE	-0.13	-0.05	2.01 ± 0.03	1.50	4.31
	–"– & lags	-0.08	-0.02	1.92 ± 0.03	1.42	4.17
	–"– & climate variables (no SSTs)	-0.02	0.05	1.86 ± 0.03	1.37	4.02
	–"– & SSTs	**0.00**	**0.05**	**1.83** ± 0.03	**1.34**	**3.94**
RF	Ens members with PE	-0.15	-0.04	2.02 ± 0.03	1.49	4.34
	–"– & lags	-0.10	0.00	1.96 ± 0.03	1.44	4.21
	–"– & climate variables (no SSTs)	-0.08	0.02	1.93 ± 0.03	1.39	4.16
	–"– & SSTs	**-0.06**	**0.04**	**1.89** ± 0.03	**1.36**	**4.08**

SSTs does not provide much additional information. In order to be consistent, we use predictions of these models trained on all features except SSTs as input to the stacking model.

2.8 Conclusion

This paper systematically explores the use of machine learning methods for subseasonal forecasting, highlighting several important factors: (1) the importance of using ensembles of physics-based forecasts (as opposed to only using the mean, as in common practice); (2) the potential for forecasting temperature and precipitation extremes using quantile regression; (3) the efficacy of different mechanisms, such as positional encoding and convolutional neural networks, for modeling spatial dependencies; (4) the importance of various features, such as sea surface temperature and lagged temperature and precipitation values, for pre-

Table 2.11: Grouped feature importance results on validation for temperature regression task using NCEP-CFSv2 ensemble members. The results demonstrate that using some additional information may yield only very small improvements in predictive skill, and in some cases, the side information may decrease predictive skill. −"− means a repetition of features that are used above. For example, in the U-Net part of the table, "−"− & lags" means that ensemble members, PE, and lags are used as features and "−"− & SSTs" means ensemble members, PE and lags, land features and SSTs are used as features. The **best** results are in bold. MSE is reported in squared $^{\circ}C$.

Model	Features	Mean R^2 ($\uparrow$)	Median R^2 ($\uparrow$)	Mean Sq Err ($\downarrow$)	Median MSE ($\downarrow$)	90th prctl MSE ($\downarrow$)
LR	Ens members	0.35	0.40	2.19 ± 0.02	2.00	3.47
	−"− & lags	**0.37**	**0.40**	**2.12 ± 0.02**	**1.94**	**3.30**
	−"− & climate variables (no SSTs)	0.36	0.39	2.14 ± 0.04	1.94	3.40
	−"− & SSTs	0.34	0.38	2.23 ± 0.02	1.99	3.73
U-Net	Ens members with PE	**0.33**	**0.41**	**2.22 ± 0.04**	**2.02**	**3.47**
	−"− & lags	0.32	0.40	2.24 ± 0.02	2.02	3.49
	−"− & climate variables (no SSTs)	0.31	0.41	2.26 ± 0.02	2.08	3.48
	−"− & SSTs	0.28	0.38	2.47 ± 0.02	2.20	3.95
RF	Ens members with PE	0.11	0.37	2.85 ± 0.04	2.28	4.87
	−"− & lags	0.30	0.36	2.35 ± 0.02	2.12	3.70
	−"− & climate variables (no SSTs)	**0.30**	**0.36**	**2.33 ± 0.02**	**2.10**	**3.65**
	−"− & SSTs	0.28	0.34	2.42 ± 0.02	2.17	3.83

dictive accuracy; (5) model stacking provides substantial benefits by leveraging the different utilization of spatial data among contributing models. The stacking model probably capitalizes on this diversity, fostering performance enhancement. Together, these results provide new insights into using ML for subseasonal weather forecasting in terms of the selection of features, models, and methods.

Our results also suggest several important directions for future research. In terms of **features**, there are many climate forecasting ensembles computed by organizations such as NOAA and ECMWF. This paper focuses on ensembles in which ensemble members have a distinct ordering (in terms of lagged initial conditions used to generate them), but other ensembles correspond to initial conditions or parameters drawn independently from some distribution. Leveraging such ensemble forecasts and potentially jointly leveraging ensemble members from multiple distinct ensembles may further improve the predictive accuracy of our methods.

In terms of **models**, new neural architecture models such as transformers show remarkable performance on several image analysis tasks [Dosovitskiy et al., 2020, Carion et al., 2020, Chen et al., 2021, Khan et al., 2022] and have potential in the context of forecasting climate temperature and precipitation maps. A careful study is needed, as past image analysis work using transformers generally uses large quantities of training data, exceeding what is available in SSF contexts. Recent advancements in data-driven global weather forecasting models, such as Pangu-Weather [Bi et al., 2022], FourCastNet [Pathak et al., 2022], and GenCast [Price et al., 2023], demonstrate the potential of ML techniques to enhance forecasting capabilities across various timescales. These models outperform traditional numerical weather prediction approaches, suggesting that similar data-driven methods may hold promise for improving SSF quality.

In terms of **methods**, two outstanding challenges are particularly salient to the SSF community. The first is uncertainty quantification; that is, we wish not only to forecast temperature or precipitation but also to predict the likelihood of certain extreme events. Our work on quantile regression is an important step in this direction and statistical methods like conformalized quantile regression [Romano et al., 2019] may provide additional insights. Second, we see in Figure A.16 that, at least in some geographic regions, the distribution of ensemble hindcast and forecast data may be quite different. Employing methods that are more robust to *distribution drift* [Wiles et al., 2021, Subbaswamy et al., 2021, Zhu et al., 2021b] is particularly important not only for handling forecast and hindcast data but also for accurate SSF in a changing climate.

CHAPTER 3

PRESERVING INVARIANT MEASURES IN CHAOTIC SYSTEMS WITH NEURAL OPERATOR TRAINING

3.1 Introduction

Training fast and accurate surrogate models to emulate complex dynamical systems is key to many scientific applications of machine learning, including climate modeling [Pathak et al., 2022], fluid dynamics [Li et al., 2021, Takamoto et al., 2022], plasma physics [Mathews et al., 2021], and molecular dynamics [Unke et al., 2021, Musaelian et al., 2023]. Fast emulators are powerful tools that can be used for forecasting and data assimilation [Chen et al., 2022, Pathak et al., 2022, Mathews et al., 2021], sampling to quantify uncertainty and compute statistical properties [Pathak et al., 2022, Unke et al., 2021, Musaelian et al., 2023], identifying latent dynamical parameters [Lu et al., 2020, Wang et al., 2022a], and solving a wide range of inverse problems [Jiang and Willett, 2022]. Specifically, neural operator architectures [Li et al., 2021, Lu et al., 2021, Li et al., 2022a] have been shown to be promising physics-informed surrogate models for emulating spatiotemporal dynamics, such as dynamics governed by partial differential equations (PDEs).

One key feature of many of the dynamical systems in these applications is chaos, which is characterized by a high sensitivity to initial conditions and results in a theoretical limit on the accuracy of forecasts. Chaotic dynamics ensure that no matter how similarly initialized, any two distinct trajectories will diverge at an exponential rate while remaining confined to a chaotic attractor [Medio and Lines, 2001]. At the same time, chaos is fundamental to many critical physical processes, such as turbulence in fluid flows [Davidson, 2015] as well as mixing and ergodicity [Medio and Lines, 2001]—properties that underpin the fundamental assumptions of statistical mechanics [Dorfman, 1999].

Chaos not only presents a barrier to accurate forecasts but also makes it challenging to train emulators, such as neural operators, using the traditional approach of rolling-out multiple time steps and fitting the root mean squared error (RMSE) of the prediction, as demonstrated in Figure 3.1. This is because RMSE training relies on encouraging the emulator to produce more and more accurate forecasts, ideally over a long time horizon, which is severely limited by the chaotic dynamics. This problem is exacerbated by measurement noise, which, in combination with the high sensitivity to initial conditions, further degrades the theoretical limit on forecasting. Unfortunately, this is precisely the setting for many real-world scientific and engineering applications. While accurate long-term forecasts are impossible in this setting, it is still possible to replicate the statistical properties of chaotic dynamics.

Specifically, we can define a natural invariant measure on a chaotic attractor that characterizes the time-invariant statistical properties of the dynamics on the attractor [Medio and Lines, 2001]. By training a neural operator to preserve this invariant measure—or equivalently, preserve time-invariant statistics—we can ensure that the neural operator is properly emulating the chaotic dynamics even though it is not able to perform accurate long-term forecasts. In this paper, we introduce two new training paradigms to address this challenge. The first uses an optimal transport-based objective to ensure the invariant measure is preserved. While effective, this approach requires expert knowledge of invariant measures to determine appropriate training losses. The second paradigm uses a contrastive feature loss that naturally preserves invariant measures without this prior expert knowledge. These new losses, which are both intended to preserve the invariant statistics that govern the long-term behavior of the dynamics, are used in combination with the standard RMSE loss evaluated over a short time horizon, which ensures any remaining predictability in the short-term dynamics is correctly captured.

We operate in the multi-environment setting, in which parameters governing the system evolution may be different for each train and test sample. This setting is more challenging than the more typical single-environment setting because it requires emulators to generalize over a broader range domain. Most practical use cases for emulators—where the computational costs of generating training data and training an emulator are outweighed by the computational gains at deployment—are in the multi-environment setting.

3.1.1 Contributions

This paper makes the following key contributions. Firstly, we identify, frame, and empirically illustrate the problem of training standard neural operators on chaotic dynamics using only RMSE—namely, that the sensitivity to initial conditions in combination with noise means that any predictive model will quickly and exponentially diverge from the true trajectory in terms of RMSE. As such, RMSE is a poor signal for training a neural operator. We instead suggest training neural operators to preserve the invariant measures of chaotic attractors and the corresponding time-invariant statistics, which are robust to the combination of noise and chaos. Secondly, we propose a direct optimal transport-based approach to train neural operators to preserve the distribution of a chosen set of summary statistics. Specifically, we use a Sinkhorn divergence loss based on the Wasserstein distance to match the distribution of the summary statistics between the model predictions and the data. Next, we propose a general-purpose contrastive learning approach to learn a set of invariant statistics directly from the data without expert knowledge. Then, we construct a loss function to train neural operators to preserve these learned invariant statistics. Finally, we empirically test both of these approaches and show that the trained neural operators capture the true invariant statistics—and therefore the underlying invariant measures of the chaotic attractors—much more accurately than baseline neural operators trained using only RMSE, resulting in more stable and physically relevant long-term predictions.

3.1.2 Related work

Neural operators. The goal of a neural operator, proposed in the context of dynamical modeling, is to approximate the semigroup relationship between the input and output function space [Li et al., 2021, Lu et al., 2021, Li et al., 2022a,b, Brandstetter et al., 2022b, Gupta and Brandstetter, 2022, Brandstetter et al., 2022c]. It is particularly effective in handling complex systems governed by partial differential equations (PDEs), where the neural operator is designed to operate on an entire function or signal. The architecture designs vary significantly in recent works, including the Fourier neural operator (FNO) which uses Fourier space convolution [Li et al., 2021], the deep operator network (DeepONet) Lu et al. [2021], which consists of two subnetworks modeling the input sensors and output locations, and other modern designs like transformers [Li et al., 2022a] and graph neural networks [Brandstetter et al., 2022c].

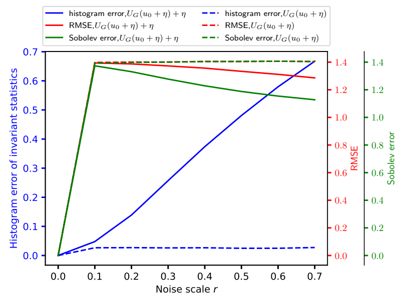
Multi-environment learning. Recent developments in multi-environment learning often involve adjusting the backbone of the neural operator. For instance, Dyad [Wang et al., 2022a] employs an encoder to extract time-invariant hidden features from dynamical systems, utilizing a supervisory regression loss relative to the context values. These hidden features are then supplied as additional inputs to the neural operator for decoding in a new environment. In contrast, Context-informed Dynamics Adaptation (CoDA, [Kirchmeyer et al., 2022]) decodes the environment context via a hypernetwork, integrating the parameters of the hypernetwork as an auxiliary input to the neural operator. Like our approach, Dyad learns time-invariant features from dynamical systems. However, their methods require a time series of measurements instead of just initial conditions. In addition, both CoDA and Dyad alter the backbone of the neural operator, making it challenging to adapt to new neural operator architectures. In contrast, our method uses the learned time-invariant features to determine a loss function that can be applied to any neural operator architecture without changes to either the backbone or the inputs.

Long-term predictions. Lu et al. [2017] try to recover long-term statistics by learning a discrete-time stochastic reduced-order system. From the perspective of training objectives, Li et al. [2022b] propose using a Sobolev norm and dissipative regularization to facilitate learning of long-term invariant measures. The Sobolev norm, which depends on higher-order derivatives of both the true process and the output of the neural operator, can capture high-frequency information and is superior to using only RMSE. However, its effectiveness diminishes in noisy and chaotic scenarios where it struggles to capture the correct statistics. The dissipative loss term, regularizes the movement of the neural operator with respect to the input, and attempts to make the emulator stay on the attractor. However, in the case of a multi-environment setup for chaotic systems, our goal is to learn a neural operator capable of modeling long-term distributions with regard to different contexts. Using the dissipative loss as a regularization will cause the neural operator to be insensitive to the context and fail to model different attractors.

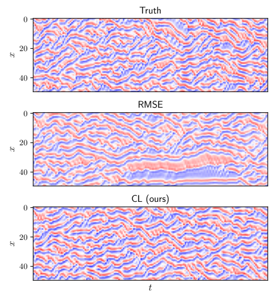
Prior [Mikhaeil et al., 2022] and concurrent work [Hess et al., 2023] on training recurrent neural networks (RNNs) have suggested that using teacher forcing methods with a squared error loss on short time sequences can produce high-quality emulators for chaotic time series. Instead, we focus on training neural operators on fully observed high-dimensional deterministic chaos and find that using only an RMSE loss generally fails to perform well on noisy chaotic dynamics. Another concurrent work by Platt et al. [2023] has suggested using dynamical invariants such as the Lyapunov spectrum and the fractal dimension to regularize training for reservoir computing emulators. We show that emulators trained using our approaches also correctly capture the Lyapunov spectrum and fractal dimension of the chaotic attractors without explicitly including these invariants, which are difficult to estimate in high dimensions, in our loss.

Wasserstein distance and optimal transport. Optimal transport theory and the Wasserstein metric have become powerful theoretical and computational tools. for a variety of applications, including generative computer vision models [Arjovsky et al., 2017], geometric machine learning and data analysis [Khamis et al., 2023], particle physics [Komiske et al., 2019], as well as identify conservation laws [Lu et al., 2023] and fitting parameterized models for low-dimensional dynamical systems [Yang et al., 2023b]. In particular, Yang et al. [2023b] uses the Wasserstein distance as an optimization objective to directly fit a parameterized model to data from a low-dimensional attractor. They compute the Wasserstein distance by turning the nonlinear dynamics in the original state space into a linear PDE in the space of distributions and then solving a PDE-constrained optimization problem. This has some nice theoretical properties but generally scales poorly to high-dimensional dynamical systems. Similarly, concurrent work by Botvinick-Greenhouse et al. [2023] suggests modeling dynamics by fitting a Fokker–Planck PDE for the probability density of the state using the Wasserstein distance. However, since this also requires estimating the full probability distribution of the state on a mesh grid, it is again challenging to scale this method to high-dimensional systems. In contrast, our proposed optimal transport approach focuses on fitting deep learning-based neural operators to high-dimensional chaotic dynamical systems by first choosing a set of physically relevant summary statistics and then using the computationally efficient Sinkhorn algorithm [Cuturi, 2013] to compute our optimal transport loss—an entropy-regularized approximation for the (squared) Wasserstein distance—on the distribution of the summary statistics.

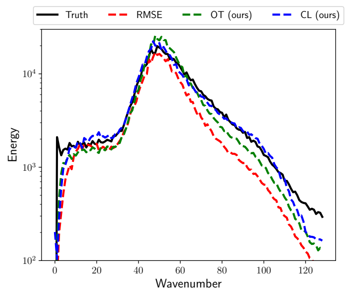
Feature loss and contrastive learning. Contrastive learning [Chen et al., 2020, He et al., 2020a, Wu et al., 2018] charges an encoder with generating features that are unaffected by transformations by promoting the similarity of features between different transformations of the same image and by maximizing the feature distance between distinct images. The recent advancements in contrastive learning mainly result from using extensive data augmentation to motivate the encoder to learn semantic representations. In addition to its success in image recognition, contrastive learning has proven to be effective in modeling fine-grained



(a) Error metrics vs. noise scale r.



(b) Predictions from emulators.



(c) Energy spectrum.

Figure 3.1: The impact of noise on invariant statistics vs. RMSE and Sobolev norm. (a) We show the impact of noise on various error metrics using ground truth simulations of the chaotic Kuramoto–Sivashinsky (KS) system with increasingly noisy initial conditions $\mathbf{U}_G(\mathbf{u}_0+\eta)$ as well as with added measurement noise $\mathbf{U}_G(\mathbf{u}_0+\eta)+\eta$. Here, $\mathbf{U}_G(\cdot)$ refers to the ground truth solution to the differential equation (3.1) for the KS system given an initial condition, and $\eta \sim \mathcal{N}(0, r^2\sigma^2 I)$, where σ^2 is the temporal variance of the trajectory $\mathbf{U}_G(\mathbf{u}_0)$ and r is the noise scale. Relative RMSE and Sobolev norm [Li et al., 2022b], which focus on short-term forecasts, deteriorate rapidly with noise η, whereas the invariant statistics have a much more gradual response to noise, indicating robustness. (b) The emulator trained with only RMSE degenerates at times into striped patches, while ours is much more statistically consistent with the ground truth. (c) Again, the emulator trained with only RMSE performs the worst in terms of capturing the expected energy spectrum over a long-term prediction.

structures [Zhang and Maire, 2020] and in scientific applications [Jiang and Willett, 2022]. Training data generators or solving inverse problems generally requires output to be coherent across multiple structural measurements, a requirement that pixel-wise MSE typically fails to meet. For this reason, Johnson et al. [2016] computes MSE on deep features of the classification models as a structural distance metric to train generative models. Tian et al. [2021] propose the encoder of contrastive learning as an unsupervised alternative for calculating feature loss.

3.2 Problem Formulation

Consider a dynamical system with state space $\mathbf{u} \in \mathcal{U}$ governed by

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = G(\mathbf{u}, \phi), \tag{3.1}$$

where G is the governing function, and $\phi \in \Phi$ is a set of parameters that specify the dynamics. We will denote trajectories governed by these dynamics (3.1) as $\mathbf{u}(t)$, particular points on the trajectory as $\mathbf{u}_t \in \mathcal{U}$, and a sequence of $K+1$ consecutive time points on the trajectory as $\mathbf{U}_{I:I+K} := \{\mathbf{u}_{t_i}\}_{i=I}^{I+K}$. Our experiments use equally spaced time points with a time step Δt. We refer to settings where ϕ varies as *multi-environment settings*.

Our goal is to learn an emulator in a multi-environment setting that approximates the dynamics (3.1) with a data-driven neural operator $\hat{g}_\theta : \mathcal{U} \times \Phi \to \mathcal{U}$ that makes discrete predictions

$$\hat{\mathbf{u}}_{t+\Delta t} := \hat{g}_\theta(\hat{\mathbf{u}}_t, \phi), \tag{3.2}$$

where θ are the parameters of the neural operator $\hat{g}_\theta$. In the multi-environment setting, we will have data from a variety of environments $n \in \{1, 2, \ldots, N\}$, and thus our training data $\{\mathbf{U}_{0:L}^{(n)}, \phi^{(n)}\}_{n=1}^{N}$ consists of trajectories $\mathbf{U}_{0:L}^{(n)}$ and the corresponding environment parameters $\phi^{(n)}$.

Predicted sequence notation. We will denote a sequence of $h+1$ autonomously predicted states (using the neural operator $\hat{g}_\theta$) with an initial condition $\mathbf{u}_{t_I}^{(n)}$ as

$$\hat{\mathbf{U}}_{I:I+h}^{(n)} := \{\mathbf{u}_{t_I}^{(n)}, \hat{g}_\theta(\mathbf{u}_{t_I}^{(n)}, \phi^{(n)}), \hat{g}_\theta \circ \hat{g}_\theta(\mathbf{u}_{t_I}^{(n)}, \phi^{(n)}), \ldots, \overbrace{\hat{g}_\theta \circ \cdots \circ \hat{g}_\theta}^{h}(\mathbf{u}_{t_I}^{(n)}, \phi^{(n)})\}. \tag{3.3}$$

We will often use a concatenated sequence (with total length $K+1$) of these autonomous prediction sequences (each with length $h+1$), which we will denote as

$$\hat{\mathbf{U}}_{I:h:I+K}^{(n)} := \hat{\mathbf{U}}_{I:I+h}^{(n)} \oplus \hat{\mathbf{U}}_{I+h+1:I+2h+1}^{(n)} \oplus \cdots \oplus \hat{\mathbf{U}}_{I+K-h:I+K}^{(n)}, \tag{3.4}$$

where $\oplus$ is concatenation.

Chaotic dynamical systems and invariant measures of chaotic attractors. We focus on chaotic dynamical systems that have one or more chaotic attractors, which exhibit a sensitive dependence on initial conditions characterized by a positive Lyapunov exponent [Medio and Lines, 2001]. For each chaotic attractor $\mathcal{A}$, we can construct a natural invariant measure (also known as a physical measure)

$$\mu_{\mathcal{A}} = \lim_{T\to\infty} \frac{1}{T} \int^{T} \delta_{\mathbf{u}_{\mathcal{A}}(t)} \, \mathrm{d}t, \tag{3.5}$$

where $\delta_{\mathbf{u}(t)}$ is the Dirac measure centered on a trajectory $\mathbf{u}_{\mathcal{A}}(t)$ that is in the basin of attraction of $\mathcal{A}$ [Medio and Lines, 2001]. Note that, because $\mathcal{A}$ is an attractor, any trajectory $\mathbf{u}_{\mathcal{A}}(t)$ in the basin of attraction of $\mathcal{A}$ will give the same invariant measure $\mu_{\mathcal{A}}$, i.e. the dynamics are ergodic on $\mathcal{A}$. Therefore, any time-invariant statistical property $S_{\mathcal{A}}$ of the dynamics on $\mathcal{A}$ can be written as

$$S_{\mathcal{A}} = \mathbb{E}_{\mu_{\mathcal{A}}}[s] = \int s(\mathbf{u}) \, \mathrm{d}\mu_{\mathcal{A}}(\mathbf{u}) = \lim_{T\to\infty} \frac{1}{T} \int^{T} s(\mathbf{u}_{\mathcal{A}}(t)) \, \mathrm{d}t \tag{3.6}$$

for some function $s(\mathbf{u})$ and a trajectory $\mathbf{u}_{\mathcal{A}}(t)$ in the basin of attraction of $\mathcal{A}$. Conversely, for any $s(\mathbf{u})$, (3.6) gives a time-invariant property $S_{\mathcal{A}}$ of the dynamics.

In this work, we assume each sampled trajectory in the data is from a chaotic attractor and therefore, has time-invariant statistical properties characterized by the natural invariant measure of the attractor.

Noisy measurements. Because of the sensitivity to initial conditions, accurate long-term forecasts (in terms of RMSE) are not possible for time scales much larger than the Lyapunov time [Medio and Lines, 2001]. This is because any amount of noise or error in the measurement or forecast model will eventually result in exponentially diverging trajectories. The noisier the data, the more quickly this becomes a problem (Figure 3.1). However, in the presence of measurement noise, invariant statistics of the noisy trajectory

$$\tilde{\mathbf{u}}_{\mathcal{A}}(t) = \mathbf{u}_{\mathcal{A}}(t) + \eta, \quad \eta \sim p_{\eta} \tag{3.7}$$

can still provide a useful prediction target. The invariant statistics will be characterized by a broadened measure (the original measure convolved with the noise distribution p_η)

$$\tilde{\mu}_{\mathcal{A}} = \lim_{T\to\infty} \frac{1}{T} \int^{T} \delta_{\tilde{\mathbf{u}}_{\mathcal{A}}(t)}\,\mathrm{d}t = \mu_{\mathcal{A}} * p_\eta = \lim_{T\to\infty} \frac{1}{T} \int^{T} p_\eta(\mathbf{u}_{\mathcal{A}}(t))\,\mathrm{d}t \tag{3.8}$$

and are therefore given by

$$\begin{aligned}\tilde{S}_{\mathcal{A}} = \mathbb{E}_{\tilde{\mu}_{\mathcal{A}}}[s] = \int s(\mathbf{u})\,\mathrm{d}\tilde{\mu}_{\mathcal{A}}(\mathbf{u}) \\ = \lim_{T\to\infty} \frac{1}{T} \int^{T} s(\tilde{\mathbf{u}}_{\mathcal{A}}(t))\,\mathrm{d}t = \lim_{T\to\infty} \frac{1}{T} \int^{T} s(\mathbf{u}_{\mathcal{A}}(t))\,p_\eta(\mathbf{u}_{\mathcal{A}}(t))\,\mathrm{d}t,\end{aligned} \tag{3.9}$$

which can be a good approximation for $S_{\mathcal{A}} = \mathbb{E}_{\mu_{\mathcal{A}}}[s]$ even in noisy conditions and does not suffer from the exponential divergence of RMSE (Figure 3.1).

3.3 Proposed Approaches

3.3.1 Physics-informed optimal transport

We propose an optimal transport-based loss function to match the distributions of a set of summary statistics $\mathbf{s}(\mathbf{u}) = \left[s^{(1)}(\mathbf{u}), s^{(2)}(\mathbf{u}), \ldots, s^{(k)}(\mathbf{u})\right]$ between the data $\mathbf{s}_i = \mathbf{s}(\mathbf{u}_{t_i}) \sim \mu_{\mathbf{s}}$ and the model predictions $\hat{\mathbf{s}}_j = \mathbf{s}(\hat{\mathbf{u}}_{t_j}) \sim \mu_{\hat{\mathbf{s}}}$. Here, $\mu_{\mathbf{s}}$ and $\mu_{\hat{\mathbf{s}}}$ are the probability measures of the summary statistics for the true chaotic attractor $\mathcal{A}$ and the attractor $\hat{\mathcal{A}}$ learned by the neural operator, respectively. Specifically, $\mu_{\mathbf{s}}(\mathbf{s}') = \int \delta_{\mathbf{s}(\mathbf{u})-\mathbf{s}'}\,\mu_{\mathcal{A}}(\mathbf{u})\,\mathrm{d}\mathbf{u}$ and $\mu_{\hat{\mathbf{s}}}(\hat{\mathbf{s}}') = \int \delta_{\mathbf{s}(\hat{\mathbf{u}})-\hat{\mathbf{s}}'}\,\mu_{\hat{\mathcal{A}}}(\hat{\mathbf{u}})\,\mathrm{d}\hat{\mathbf{u}}$. By choosing the set of summary statistics using expert domain knowledge, this approach allows us to directly guide the neural operator toward preserving important physical properties of the system.

Optimal transport—Wasserstein distance. We match the distributions of summary statistics using the Wasserstein distance $W(\mu_{\mathbf{s}}, \mu_{\hat{\mathbf{s}}})$—a distance metric between probability

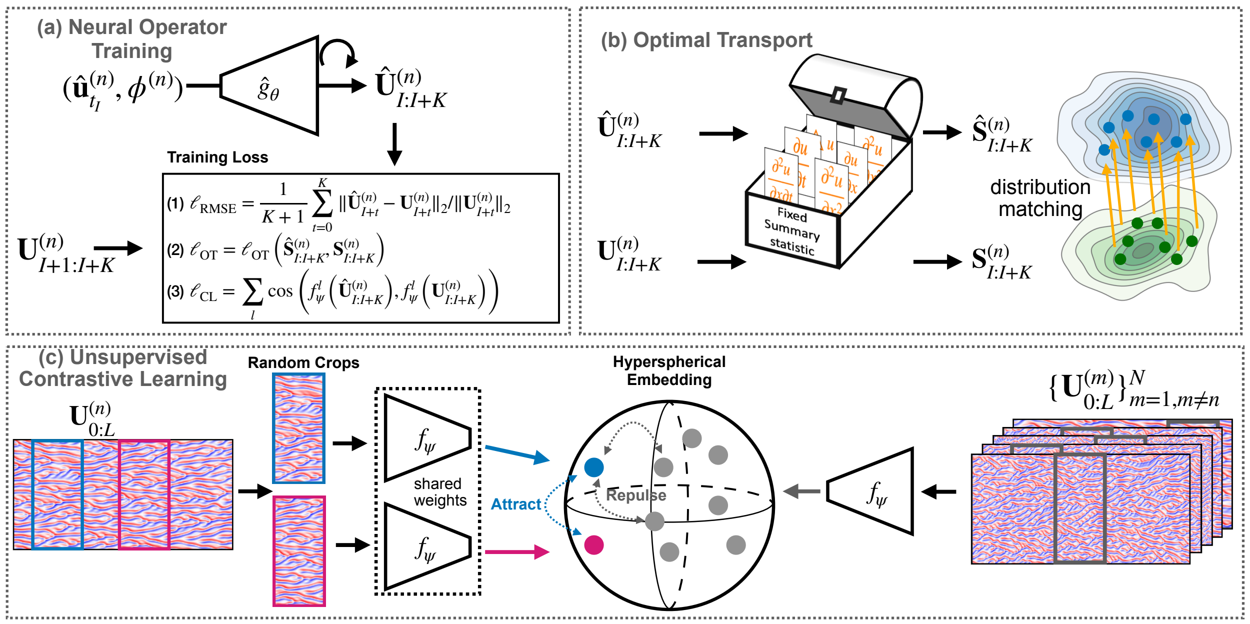

(a) Neural Operator Training
$(\hat{\mathbf{u}}^{(n)}_{t_I}, \phi^{(n)})$
$\hat{g}_\theta$
$\hat{\mathbf{U}}^{(n)}_{I:I+K}$
$\mathbf{U}^{(n)}_{I+1:I+K}$
Training Loss
(1) $\ell_{\text{RMSE}} = \frac{1}{K+1}\sum_{t=0}^{K} \|\hat{\mathbf{U}}^{(n)}_{I+t} - \mathbf{U}^{(n)}_{I+t}\|_2 / \|\mathbf{U}^{(n)}_{I+t}\|_2$
(2) $\ell_{\text{OT}} = \ell_{\text{OT}}\left(\hat{\mathbf{S}}^{(n)}_{I:I+K}, \mathbf{S}^{(n)}_{I:I+K}\right)$
(3) $\ell_{\text{CL}} = \sum_l \cos\left(f^l_\psi\left(\hat{\mathbf{U}}^{(n)}_{I:I+K}\right), f^l_\psi\left(\mathbf{U}^{(n)}_{I:I+K}\right)\right)$
(b) Optimal Transport
$\hat{\mathbf{U}}^{(n)}_{I:I+K}$
$\mathbf{U}^{(n)}_{I:I+K}$
Fixed Summary statistic
$\hat{\mathbf{S}}^{(n)}_{I:I+K}$
$\mathbf{S}^{(n)}_{I:I+K}$
distribution matching
(c) Unsupervised Contrastive Learning
$\mathbf{U}^{(n)}_{0:L}$
Random Crops
f_ψ
shared weights
Attract
Repulse
Hyperspherical Embedding
$\{\mathbf{U}^{(m)}_{0:L}\}^N_{m=1,m\neq n}$

Figure 3.2: Our proposed approaches for training neural operators. (a) Neural operators are emulators trained to take an initial state and output future states in a recurrent fashion. To ensure the neural operator respects the statistical properties of chaotic dynamics when trained on noisy data, we propose two additional loss functions for matching relevant long-term statistics. (b) We match the distribution of summary statistics, chosen based on prior knowledge, between the emulator predictions and noisy data using an optimal transport loss. (c) In the absence of prior knowledge, we take advantage of self-supervised contrastive learning to automatically learn relevant time-invariant statistics, which can then be used to train neural operators.

measures defined in terms of the solution to an optimal transport problem:

$$\frac{1}{2}W(\mu_{\mathbf{s}}, \mu_{\hat{\mathbf{s}}})^2 := \inf_{\pi \in \Pi(\mu_{\mathbf{s}}, \mu_{\hat{\mathbf{s}}})} \int c(\mathbf{s}, \hat{\mathbf{s}})\, \mathrm{d}\pi(\mathbf{s}, \hat{\mathbf{s}}). \tag{3.10}$$

We use a quadratic cost function $c(\mathbf{s}, \hat{\mathbf{s}}) = \frac{1}{2}\|\mathbf{s} - \hat{\mathbf{s}}\|^2$ (i.e., W is the 2-Wasserstein distance), and the map $\pi \in \Pi(\mu_{\mathbf{s}}, \mu_{\hat{\mathbf{s}}})$ must be a valid transport map between $\mu_{\mathbf{s}}$ and $\mu_{\hat{\mathbf{s}}}$ (i.e., a joint distribution for $\mathbf{s}, \hat{\mathbf{s}}$ with marginals that match $\mu_{\mathbf{s}}$ and $\mu_{\hat{\mathbf{s}}}$) [Villani, 2009].

In the discrete setting with distributions represented by samples $\mathbf{S} = \{\mathbf{s}_i\}_{i=1}^{L}$ and $\hat{\mathbf{S}} =$

$\{\hat{\mathbf{s}}_j\}_{j=1}^L$, the Wasserstein distance is given by

$$\frac{1}{2}W(\mathbf{S},\hat{\mathbf{S}})^2 := \min_{T\in\Pi}\sum_{i,j} T_{ij}C_{ij}, \tag{3.11}$$

where the cost matrix $C_{ij} = \frac{1}{2}\|\mathbf{s}_i - \hat{\mathbf{s}}_j\|^2$. The set of valid discrete transport maps Π consists of all matrices T such that $\forall i,j$, $T_{ij} \geq 0$, $\sum_j T_{ij} = 1$, and $\sum_i T_{ij} = 1$.

Entropy-regularized optimal transport—Sinkhorn divergence. Exactly solving the optimal transport problem associated with computing the Wasserstein distance is computationally prohibitive, especially in higher dimensions. A common approximation made to speed up computation is to introduce an entropy regularization term, resulting in a convex relaxation of the original optimal transport problem:

$$\frac{1}{2}W^\gamma(\mathbf{S},\hat{\mathbf{S}})^2 := \min_{T\in\Pi}\sum_{i,j} T_{ij}C_{ij} - \gamma\, h(T), \tag{3.12}$$

where $h(T) = -\sum_{i,j} T_{ij}\log T_{ij}$ is the entropy of the transport map. This entropy-regularized optimal transport problem can be solved efficiently using the Sinkhorn algorithm [Cuturi, 2013].

As $\gamma \to 0$, the entropy-regularized Wasserstein distance $W^\gamma \to W^0 = W$ reduces to the exact Wasserstein distance (3.11). For $\gamma > 0$, we can further correct for an entropic bias to obtain the Sinkhorn divergence [Feydy et al., 2019, Janati et al., 2020]

$$\ell_{\text{OT}}(\mathbf{S},\hat{\mathbf{S}}) = \frac{1}{2}\overline{W}^\gamma(\mathbf{S},\hat{\mathbf{S}})^2 := \frac{1}{2}\left(W^\gamma(\mathbf{S},\hat{\mathbf{S}})^2 - \frac{W^\gamma(\mathbf{S},\mathbf{S})^2 + W^\gamma(\hat{\mathbf{S}},\hat{\mathbf{S}})^2}{2}\right), \tag{3.13}$$

which gives us our optimal transport loss. Combined with relative root mean squared error (RMSE)

$$\ell_{\text{RMSE}}(\mathbf{U},\hat{\mathbf{U}}) := \frac{1}{K+1}\sum_{\mathbf{u}_t,\hat{\mathbf{u}}_t\in\mathbf{U},\hat{\mathbf{U}}} \frac{\|\mathbf{u}_t - \hat{\mathbf{u}}_t\|_2}{\|\mathbf{u}_t\|_2} \tag{3.14}$$

for short-term prediction consistency [Li et al., 2021, 2022b, Gupta and Brandstetter, 2022], our final loss function is

$$\ell(\theta) = \mathop{\mathbb{E}}_{\substack{n\in\{1,\ldots,N\}\\ I\in\{0,\ldots,L-K\}}} \left[\alpha\, \ell_{\mathrm{OT}}\big(\mathbf{S}^{(n)}_{I:I+K}, \hat{\mathbf{S}}^{(n)}_{I:h:I+K}\big) + \ell_{\mathrm{RMSE}}\big(\mathbf{U}^{(n)}_{I:I+K}, \hat{\mathbf{U}}^{(n)}_{I:h_{\mathrm{RMSE}}:I+K}\big)\right], \quad (3.15)$$

where $\mathbf{S}^{(n)}_{I:I+K} := \{\mathbf{s}(\mathbf{u}) \mid \mathbf{u} \in \mathbf{U}^{(n)}_{I,I+K}\}$, $\hat{\mathbf{S}}^{(n)}_{I:h:I+K} := \{\mathbf{s}(\hat{\mathbf{u}}) \mid \hat{\mathbf{u}} \in \hat{\mathbf{U}}^{(n)}_{I:h:I+K}\}$, and $\alpha > 0$ is a hyperparameter. Note that $\hat{\mathbf{S}}^{(n)}_{I:h:I+K}$ and $\hat{\mathbf{U}}^{(n)}_{I:h_{\mathrm{RMSE}}:I+K}$ implicitly depend on weights θ.

3.3.2 Contrastive feature learning

When there is an absence of prior knowledge pertaining to the underlying dynamical system, or when the statistical attributes are not easily differentiable, we propose an alternative contrastive learning-based approach to learn the relevant invariant statistics directly from the data. We first use contrastive learning to train an encoder to capture invariant statistics of the dynamics in the multi-environment setting. We then leverage the feature map derived from this encoder to construct a feature loss that preserves the learned invariant statistics during neural operator training.

Contrastive learning. The objective of self-supervised learning is to train an encoder $f_\psi(\mathbf{U})$ (with parameters ψ) to compute relevant invariant statistics of the dynamics from sequences $\mathbf{U}$ with fixed length $K+1$. We do not explicitly train on the environment parameters ϕ but rather use a general-purpose contrastive learning approach that encourages the encoder f_ψ to learn a variety of time-invariant features that are able to distinguish between sequences from different trajectories (and therefore different ϕ).

A contrastive learning framework using the Noise Contrastive Estimation (InfoNCE) loss has been shown to preserve context-aware information by training to match sets of positive pairs while treating all other combinations as negative pairs [Chen et al., 2020]. The selection of positive pairs is pivotal to the success of contrastive learning. In our approach, the key premise is that two sequences $\mathbf{U}^{(n)}_{I:I+K}$, $\mathbf{U}^{(n)}_{J:J+K}$ from the same trajectory $\mathbf{U}^{(n)}_{0:L}$ both sample the same chaotic attractor, i.e. their statistics should be similar, so we treat any such pair of sequences as positive pairs. Two sequences $\mathbf{U}^{(n)}_{I:I+K}$, $\mathbf{U}^{(m)}_{H:H+K}$ from different trajectories are treated as negative pairs. This allows us to formulate the InfoNCE loss as:

$$\ell_{\text{InfoNCE}}(\psi;\tau) := \mathop{\mathbb{E}}_{\substack{n\in\{1,\ldots,N\}\\ I,J\in\{0,\ldots,L-K\}}}\left[-\log\left(\frac{\exp\left(\langle f_\psi(\mathbf{U}^{(n)}_{I:I+K}), f_\psi(\mathbf{U}^{(n)}_{J:J+K})\rangle/\tau\right)}{\mathop{\mathbb{E}}_{\substack{m\neq n\\ H\in\{0,\ldots,L-K\}}}\left[\exp\left(\langle f_\psi(\mathbf{U}^{(n)}_{I:I+K}), f_\psi(\mathbf{U}^{(m)}_{H:H+K})\rangle/\tau\right)\right]}\right)\right]. \tag{3.16}$$

The term in the numerator enforces alignment of the positive pairs which ensures we obtain time-invariant statistics, while the term in the numerator encourages uniformity, i.e. maximizing mutual information between the data and the embedding, which ensures we can distinguish between negative pairs from different trajectories [Wang and Isola, 2020]. This provides intuition for why our learned encoder f_ψ identifies relevant time-invariant statistics that can distinguish different chaotic attractors.

Contrastive feature loss. To construct our feature loss, we use the cosine distance between a series of features of the encoder network f_ψ [Zhang et al., 2018]:

$$\ell_{\text{CL}}(\mathbf{U}, \hat{\mathbf{U}}; f_\psi) := \sum_l \cos\left(f^l_\psi(\mathbf{U}), f^l_\psi(\hat{\mathbf{U}})\right), \tag{3.17}$$

where f^l_ψ gives the output the l-th layer of the neural network. The combined loss that we use for training the neural operator is given by

$$\ell(\theta) = \mathop{\mathbb{E}}_{\substack{n\in\{1,\dots,N\}\\ I\in\{0,\dots,L-K\}}} \left[\lambda\, \ell_{\mathrm{CL}}\big(\mathbf{U}^{(n)}_{I:I+K}, \hat{\mathbf{U}}^{(n)}_{I:h:I+K}; f_\psi\big) + \ell_{\mathrm{RMSE}}\big(\mathbf{U}^{(n)}_{I:I+K}, \hat{\mathbf{U}}^{(n)}_{I:h_{\mathrm{RMSE}}:I+K}\big)\right], \tag{3.18}$$

where $\lambda > 0$ is a hyperparameter.

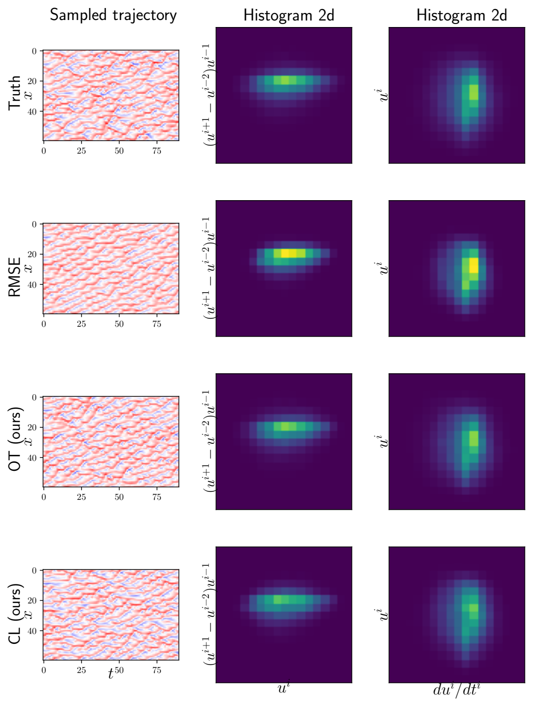
3.4 Experiments

We evaluate our approach on the 1D chaotic Kuramoto–Sivanshinsky (KS) system and a finite-dimensional Lorenz 96 system. In all cases, we ensure that the systems under investigation remain in chaotic regimes. We demonstrate the effectiveness of our approach in preserving key statistics in these unpredictable systems, showcasing our ability to handle the complex nature of chaotic systems. The code is available at: https://github.com/roxie62/neural_operators_for_chaos.

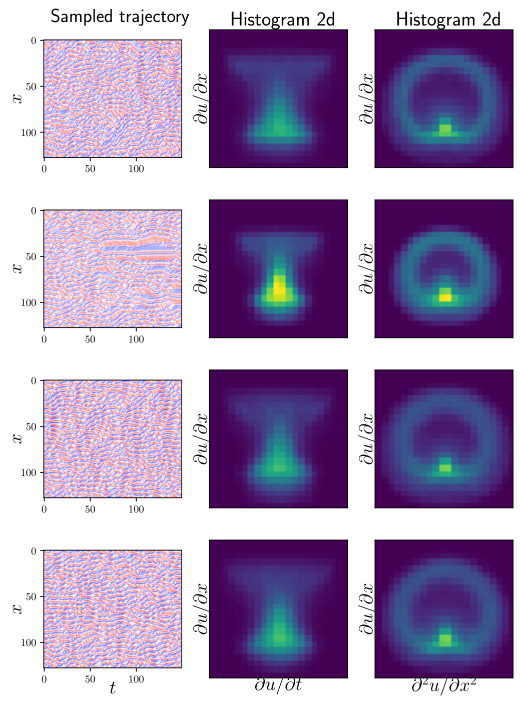
Experimental setup. Our data consists of noisy observations $\mathbf{u}(t)$ with noise $\eta \sim \mathcal{N}(0, r^2\sigma^2 I)$, where σ^2 is the temporal variance of the trajectory and r is a scaling factor. **Baselines.** We primarily consider the baseline as training with RMSE [Li et al., 2021]. We have additional baselines in Appendix B.2, including Gaussian denoising and a Sobolev norm loss with dissipative regularization [Li et al., 2022b]. **Backbones.** We use the Fourier neural operator (FNO, [Gupta and Brandstetter, 2022]). **Evaluation metrics.** We use a variety of statistics-based evaluation metrics and other measures that characterize the chaotic attractor. See Appendix B.3.1 for details.

3.4.1 Lorenz-96

As is a common test model for climate systems, data assimilation, and other geophysical applications [van Kekem, 2018, Law et al., 2016, Majda and Harlim, 2012], the Lorenz-96



(a) Lorenz-96



(b) Kuramoto–Sivashinsky

Figure 3.3: Sampled emulator dynamics and summary statistic distributions. We evaluate our proposed approaches by comparing them to a baseline model that is trained solely using relative RMSE loss. We conduct this comparison on two dynamical systems: (a) Lorenz-96 and (b) Kuramoto–Sivashinsky (KS). For each system, we show a visual comparison of the predicted dynamics (left) and two-dimensional histograms of relevant statistics (middle and right). We observe that training the neural operator with our proposed optimal transport (OT) or contrastive learning (CL) loss significantly enhances the long-term statistical properties of the emulator, as seen in the raw emulator dynamics and summary statistic distributions. The performance of the CL loss, which uses no prior knowledge, is comparable to that of the OT loss, which requires an explicit choice of summary statistics.

system is a key tool for studying chaos theory, turbulence, and nonlinear dynamical systems. It is described by the differential equation

$$\frac{du_i}{dt} = (u_{i+1} - u_{i-2})u_{i-1} - u_i + F \tag{3.19}$$

Its dynamics exhibit strong energy-conserving non-linearity, and for a large $F \geq 10$, it can exhibit strong chaotic turbulence and symbolizes the inherent unpredictability of the Earth's climate.

Experimental setup. When using optimal transport loss, we assume that expert knowledge is derived from the underlying equation. For Lorenz-96, we define the relevant statistics as $\mathbf{s}(\mathbf{u}) := \{\frac{du_i}{dt}, (u_{i+1} - u_{i-2})u_{i-1}, u_i\}$. We generate 2000 training data points with each $\phi^{(n)}$ randomly sampled from a uniform distribution with the range $[10.0, 18.0]$. We vary the noise level r from 0.1 to 0.3 and show consistent improvement in the relevant statistics.

Results. The results are presented in Table 3.1, and predictions and invariant statistics are shown in Figure 3.3 (refer to B.3.4 for more visualizations).

Table 3.1: Emulator performance on Lorenz-96 data with varying noise scale $r = 0.1, 0.2, 0.3$. The median (25th, 75th percentile) of the evaluation metrics (Appendix B.3.1) are computed on 200 Lorenz-96 test instances (each with 1500 time steps) for the neural operator trained with (1) only RMSE loss ℓ_{RMSE}; (2) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$ (using prior knowledge to choose summary statistics); and (3) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$ (without prior knowledge). We show significant improvements on the long-term statistical metrics including L_1 histogram error of the chosen statistics $\mathbf{S}(\mathbf{u}) := \{\frac{du_i}{dt}, (u_{i+1} - u_{i-2})u_{i-1}, u_i\}$; relative error of Fourier energy spectrum; and absolute error of estimated fractal dimension (FD). For high noise, OT and CL training also improve the leading Lyapunov exponent (LE) of the neural operator.

r	Training	Histogram Error ↓	Energy Spec. Error ↓	Leading LE Error ↓	FD Error ↓
0.1	ℓ_{RMSE}	0.056 (0.051, 0.062)	0.083 (0.078, 0.090)	**0.013** (0.006, 0.021)	1.566 (0.797, 2.309)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.029** (0.027, 0.032)	**0.058** (0.052, 0.064)	0.050 (0.040, 0.059)	1.424 (0.646, 2.315)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.033 (0.029, 0.037)	**0.058** (0.049, 0.065)	0.065 (0.058, 0.073)	**1.042** (0.522, 1.685)
0.2	ℓ_{RMSE}	0.130 (0.118, 0.142)	0.182 (0.172, 0.188)	0.170 (0.156, 0.191)	2.481 (1.428, 3.807)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.039** (0.035, 0.042)	**0.086** (0.079, 0.095)	0.016 (0.006, 0.030)	2.403 (1.433, 3.768)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.073 (0.066, 0.080)	0.131 (0.117, 0.149)	**0.012** (0.006, 0.018)	**1.681** (0.656, 2.682)
0.3	ℓ_{RMSE}	0.215 (0.204, 0.234)	0.291 (0.280, 0.305)	0.440 (0.425, 0.463)	3.580 (2.333, 4.866)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.057** (0.052, 0.064)	**0.123** (0.116, 0.135)	0.084 (0.062, 0.134)	3.453 (2.457, 4.782)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.132 (0.111, 0.151)	0.241 (0.208, 0.285)	**0.064** (0.045, 0.091)	**1.894** (0.942, 3.108)

3.4.2 Kuramoto–Sivashinsky

Known as a model for spatiotemporal chaos, Kuramoto–Sivashinsky (KS) has been widely used to describe various physical phenomena, including fluid flows in pipes, plasma physics, and dynamics of certain chemical reactions [Hyman and Nicolaenko, 1986]. It captures wave steepening via the nonlinear term $u\frac{\partial u}{\partial x}$, models dispersion effects through $\frac{\partial^2 u}{\partial x^2}$, and manages discontinuities by introducing hyper-viscosity via $\frac{\partial^4 u}{\partial x^4}$:

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - \phi\frac{\partial^2 u}{\partial x^2} - \frac{\partial^4 u}{\partial x^4}. \tag{3.20}$$

Experimental setup. For the KS system, we define $\mathbf{s}(\mathbf{u}) := \left\{\frac{\partial u}{\partial t}, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right\}$ [Vlachas et al., 2022]. We generate 2000 training data points, with each $\phi^{(n)}$ being randomly selected from a uniform distribution within the range of $[1.0, 2.6]$. **Results.** We report our results over 200 test instances in Table 3.2 and visualize the predictions and invariant statistics in Figure 3.3.

Table 3.2: Emulator performance on Kuramoto–Sivashinsky data with noise scale $r = 0.3$. The median (25th, 75th percentile) of the evaluation metrics (Appendix B.3.1) are computed on 200 Kuramoto–Sivashinsky test instances (each with 1000 time steps) for the neural operator trained with (1) only RMSE loss ℓ_{RMSE}; (2) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$ (using prior knowledge to choose summary statistics); and (3) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$ (without prior knowledge). We again show significant improvements in the long-term statistical metrics including L_1 histogram error of the chosen statistics $\mathbf{S}(\mathbf{u}) := \{\frac{\partial u}{\partial t}, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\}$ and relative error of Fourier energy spectrum. The fractal dimension (FD) is highly unstable in high dimensions [Greenside et al., 1982] and could not be estimated for this dataset.

Training	Histogram Error ↓	Energy Spec. Error ↓	Leading LE Error ↓
ℓ_{RMSE}	0.390 (0.325, 0.556)	0.290 (0.225, 0.402)	0.101 (0.069, 0.122)
$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.172** (0.146, 0.197)	0.211 (0.188, 0.250)	**0.094** (0.041, 0.127)
$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.193 (0.148, 0.247)	**0.176** (0.130, 0.245)	0.108 (0.068, 0.132)

3.5 Conclusion

We have demonstrated two approaches for training neural operators on chaotic dynamics by preserving the invariant measures of the chaotic attractors. The optimal transport approach uses knowledge of the underlying system to construct a physics-informed objective that matches relevant summary statistics, while the contrastive learning approach uses a multi-environment setting to directly learn relevant invariant statistics. In both cases, we see a significant improvement in the ability of the emulator to reproduce the invariant statistics of chaotic attractors from noisy data when compared with traditional RMSE-only training that focuses on short-term forecasting.

We also find, in high-noise settings, that both of our approaches give similar or lower leading LE errors than an emulator trained on RMSE loss alone, despite the fact that our method is only encouraged to match invariant statistics of the final attractor rather than dynamical quantities. We also see evidence that the fractal dimension of the contrastive learning approach is closer to the true attractor. However, we note that the fractal dimension is difficult to reliably estimate for high-dimensional chaotic attractors [Greenside et al., 1982].

Limitations Because we rely on invariant measures, our current approach is limited to trajectory data from attractors, i.e. we assume that the dynamics have reached an attractor and are not in a transient phase. We also cannot handle explicit time dependence, including time-dependent forcing or control parameters. For the optimal transport approach, choosing informative summary statistics based on prior knowledge is key to good performance (Appendix B.2.3). For the contrastive learning approach, the quality of the learned invariant statistics also depends on the diversity of the environments present in the multi-environment setting, although our additional experiments show that we can still obtain good performance even with minimal environment diversity (Appendix B.2.4).

Future work In the future, we may be able to adapt our approaches to allow for mild time dependence by restricting the time range over which we compute statistics and select positive pairs. This would allow us to study slowly varying dynamics as well as sharp discrete transitions such as tipping points. We can also improve the diversity of the data for contrastive learning by designing new augmentations or using the training trajectory of the neural operator to generate more diverse negative pairs. We will investigate generalizing our approaches to other difficult systems, such as stochastic differential equations or stochastic PDEs, and we would like to further study the trade-offs and synergies between focusing on short-term forecasting (RMSE) and capturing long-term behavior (invariant statistics). In addition, we would like to investigate and compare training methods by Mikhaeil et al. [2022], Hess et al. [2023], Platt et al. [2023] across different architectures.

CHAPTER 4
DEEP STOCHASTIC MECHANICS FOR BOSONS

4.1 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 and Taylor IV, 1998, Liu et al., 2013], and quantum computing [Grover, 2001, Papageorgiou and 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 and 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, 1966a, 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 and 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].

4.1.1 Problem formulation

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

$$i\hbar \partial_t \psi(x,t) = \mathcal{H}\psi(x,t), \tag{4.1}$$

$$\psi(x,0) = \psi_0(x), \tag{4.2}$$

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 $\mathcal{M} = \mathbb{R}^d$ 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 C.1.

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,

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

leading to the favorable scaling properties of our approach.

4.1.2 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 (4.1). 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., 2019b], FermiNet [Pfau et al., 2020a], and PauliNet [Hermann et al., 2020] reformulates the problem (4.1) 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 and 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 and Troyer, 2017, Schmitt and 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 and 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.

4.1.3 Contributions

We are inspired by works of Nelson [1966a, 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(4.1), 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, 1966a, 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 C.5.4.

We highlight the main contributions of this work as follows:

- We propose to use a stochastic formulation of quantum mechanics [Nelson, 1966a, 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 4.2.3 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.

Table 4.1 compares properties of methods for solving Equation (4.1). 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}.

Table 4.1: 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., 2020a]	$\mathbb{R}^d$	✗	✓	$\mathcal{O}(H_d d^3)$	$\geq \mathcal{O}(H_d d^3 \text{poly}(\varepsilon^{-1}))$
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)	$\mathbb{R}^d$	✓	✓	$\mathcal{O}(Nd^2)$	$\geq \mathcal{O}(Nd^2 \text{poly}(\varepsilon^{-1}))$

4.2 Deep Stochastic Mechanics

There is a family of diffusion processes that are equivalent to Equation (4.1) in a sense that all time-marginals of any such process coincide with $|\psi(x,t)|^2$; we refer to Appendix C.5 for derivation. Assuming $\psi(x,t) = \sqrt{\rho(x,t)}e^{iS(x,t)}$, we define:

$$\begin{aligned} v(x,t) &= \frac{\hbar}{m}\nabla S(x,t), \\ u(x,t) &= \frac{\hbar}{2m}\nabla \log \rho(x,t). \end{aligned} \tag{4.3}$$

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, 1966a]:

$$\begin{aligned} \mathrm{d}Y(t) &= (v(Y(t),t) + \nu u(Y(t),t))\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\overrightarrow{W}, \\ Y(0) &\sim \left|\psi_0\right|^2, \end{aligned} \tag{4.4}$$

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

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

instead aim at approximating them with the process defined using neural network approximations v_θ, u_θ:

$$\begin{aligned} \mathrm{d}X(t) &= (v_\theta(X(t),t) + \nu u_\theta(X(t),t))\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\overrightarrow{W}, \\ X(0) &\sim \left|\psi_0\right|^2. \end{aligned} \tag{4.5}$$

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

$$X_{i+1} = X_i + (v_\theta(X_i,t_i) + \nu u_\theta(X_i,t_i))\epsilon + \mathcal{N}\big(0, \frac{\nu\hbar}{m}\epsilon I_d\big), \tag{4.6}$$

where $\epsilon > 0$ denotes a step size, $0 \le i < \frac{T}{\epsilon}$, 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 and Platen, 1992], can potentially enhance efficiency and stability when ϵ is larger.

The diffusion process from Equation (4.4) achieves sampling from $\rho = \left|\psi(x,t)\right|^2$ 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:

$$\partial_t v = -\frac{1}{m}\nabla V + \langle u, \nabla\rangle u - \langle v, \nabla\rangle v + \frac{\hbar}{2m}\nabla\langle\nabla, u\rangle, \tag{4.7a}$$

$$\partial_t u = -\nabla\langle v, u\rangle - \frac{\hbar}{2m}\nabla\langle\nabla, v\rangle, \tag{4.7b}$$

$$v_0(x) = \frac{\hbar}{m}\nabla S_0(x), \ u_0(x) = \frac{\hbar}{2m}\nabla\log\rho_0(x). \tag{4.7c}$$

These equations are derived in Appendix C.5.1 and are equivalent to the Schrödinger equation. As mentioned, our equations differ from the canonical ones developed in Nelson [1966a], Guerra [1995]. In particular, the original formulation from Equation (C.10), which we call the *Nelsonian version*, includes the Laplacian of u; in contrast, *our version* in (4.7a) 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 4.2.1.

4.2.1 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 (4.7a), (4.7b), and the third one enforces the initial conditions (4.7c). We define non-linear differential operators that appear in Equation (4.7a), (4.7b):

$$\mathcal{D}_u[v,u,x,t] = -\nabla\langle v(x,t),u(x,t)\rangle - \frac{\hbar}{2m}\nabla\langle\nabla, v(x,t)\rangle, \tag{4.8}$$

$$\begin{aligned}\mathcal{D}_v[v,u,x,t] = &-\frac{1}{m}\nabla V(x,t) + \frac{1}{2}\nabla\|u(x,t)\|^2 \\ &-\frac{1}{2}\nabla\|v(x,t)\|^2 + \frac{\hbar}{2m}\nabla\langle\nabla, u(x,t)\rangle.\end{aligned} \tag{4.9}$$

We aim to minimize the following losses:

$$\begin{aligned}L_1(v_\theta,u_\theta) = \int_0^T \mathbb{E}^X \Big\|\partial_t u_\theta(X(t),t) \\ - \mathcal{D}_u[v_\theta,u_\theta,X(t),t]\Big\|^2 \mathrm{d}t,\end{aligned} \tag{4.10}$$

$$\begin{aligned}L_2(v_\theta,u_\theta) = \int_0^T \mathbb{E}^X \Big\|\partial_t v_\theta(X(t),t) \\ - \mathcal{D}_v[v_\theta,u_\theta,X(t),t]\Big\|^2 \mathrm{d}t,\end{aligned} \tag{4.11}$$

$$L_3(v_\theta,u_\theta) = \mathbb{E}^X\|u_\theta(X(0),0) - u_0(X(0))\|^2, \tag{4.12}$$

$$L_4(v_\theta,u_\theta) = \mathbb{E}^X\|v_\theta(X(0),0) - v_0(X(0))\|^2, \tag{4.13}$$

where u_0, v_0 are defined in Equation (4.7c). Finally, we define a combined loss using a weighted sum with $w_i > 0$:

$$\mathcal{L}(\theta) = \sum_{i=1}^{4} w_i L_i(v_\theta, u_\theta). \tag{4.14}$$

The basic idea of our approach is to sample new trajectories using Equation (4.6) with $\nu = 1$ for each iteration τ. These trajectories are then used to compute stochastic estimates of the loss from Equation (4.14), 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 4.1; a more detailed version is given in Appendix C.2.

Algorithm 1 Training algorithm pseudocode

```
Input $\psi_0$ – initial wave-function, $M$ – epoch number, $B$ – batch size, other parameters (optimizer parameters, physical constants, Euler–Maruyama parameters; see Appendix C.2)
Initialize NNs $u_{\theta_0}$, $v_{\theta_0}$
for each iteration $0 \le \tau < M$ do
    Sample $B$ trajectories using $u_{\theta_\tau}, v_{\theta_\tau}$ via Equation (4.6) with $\nu = 1$
    Estimate loss $\mathcal{L}(v_{\theta_\tau}, u_{\theta_\tau})$ from Equation (4.14) 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 $\nu \ge 0$ given $X_0 \sim \mathcal{N}(0, I_d)$:

$$X_{i+1} = X_i + (v_{\theta_M}(X_i, t_i) + \nu u_{\theta_M}(X_i, t_i))\epsilon \\ + \mathcal{N}\big(0, \frac{\hbar}{m}\nu\epsilon I_d\big). \tag{4.15}$$

Appendix C.7 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 (4.7a), (4.7b), that approach would

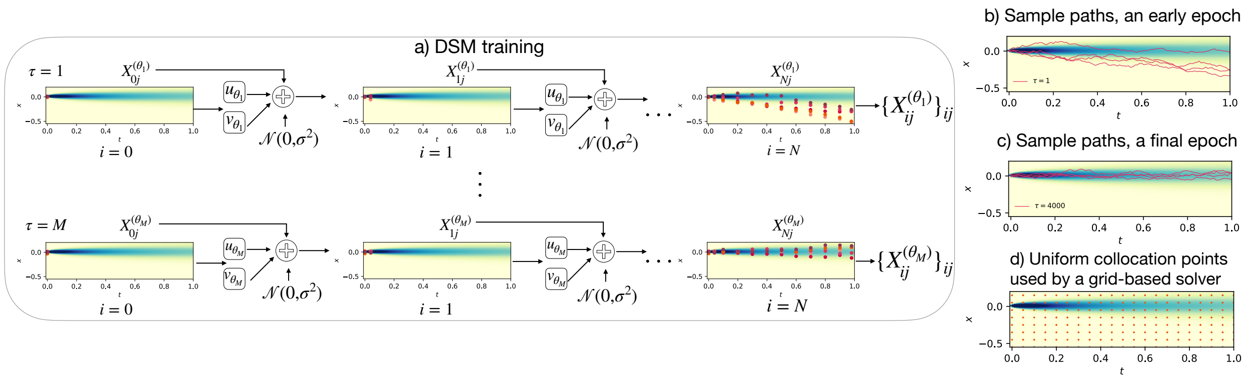


Figure 4.1: An illustration of our approach. Blue regions in the plots correspond to higher-density regions. (a) DSM training scheme: at every epoch τ, we generate B full trajectories $\{X_{ij}\}_{ij}$, $i = 0, ..., N$, $j = 1, ..., B$. Then, we update the weights of our NNs. (b) An illustration of sampled 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 $\psi(x, t)$.

suffer from having fixed sampled density (see Section 4.3). Our method, much like PINNs, seeks to minimize the residuals of the PDEs from Equations (4.7a) and (4.7b). 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.

4.2.2 Algorithmic complexity

Our formulation of stochastic mechanics with novel Equations (4.7) 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 4.2.1. The algorithmic complexity w.r.t. d of computing differential operators from Equations (4.8), (4.9) for velocities u, v is $\mathcal{O}(d^2)$. [3]

This remark is proved in Appendix C.5.5. 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.

4.2.3 Theoretical guarantees

To further justify the effectiveness of our loss function, we prove the following theorem in Appendix C.6:

Theorem 4.2.2. *(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):*

$$\sup_{t \le T} \mathbb{E}\|X(t) - Y(t)\|^2 \le C_T \mathcal{L}(v_\theta, u_\theta), \tag{4.16}$$

where constant C_T is defined explicitly in C.6.13.

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

3. Estimation of the 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.

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].

4.3 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 (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.3. Additional experiment results are given in Appendix C.4. 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 $\mu(\mathrm{d}x, \mathrm{d}t) = |\psi(x,t)|^2 \mathrm{d}x\mathrm{d}t$.

4.3.1 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

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

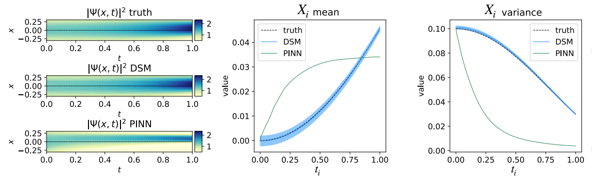
$u_0(x) = -\frac{\hbar x}{2m\sigma^2}$, $v_0(x) \equiv 0$. $X(0)$ comes from $X(0) \sim \mathcal{N}(0, \sigma^2)$, where $\sigma^2 = 0.1$.

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 $t \in [0, 1]$.

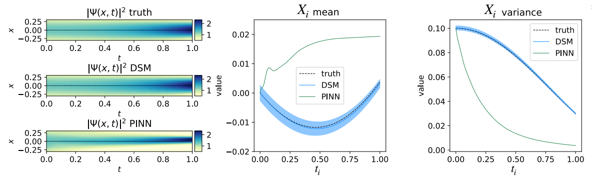
Figure 4.2(a) summarizes the results of our experiment. The left panel of the figure illustrates the evolution of the density $|\psi(x,t)|^2$ over time for different methods. It is evident that our approach accurately captures the density evolution, while the PINN model initially aligns with the ground 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 (4.1); we illustrate this effect in Figure 4.1 (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 4.2 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 4.2 (b) and Table 4.2 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.

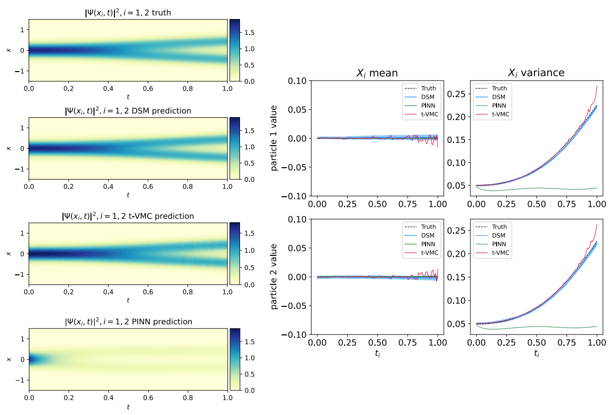
In addition, we consider an oscillator model with three non-interacting particles, which



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 4.2: 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.

can be seen as a 3d system. The results are given in Table 4.2 and Appendix C.4.2.

Table 4.2: Results for different harmonic oscillator settings. In the 3d setting, the reported errors are averaged across all dimensions. The **best** results are in bold (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.)

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	$2.9 \times 10^{-3} \pm 2.4 \times 10^{-4}$	$3.5 \times 10^{-4} \pm 0.8 \times 10^{-4}$

4.3.2 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 (4.14) 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 4.2 shows worse quantitative results compared to our approach in the setting from Section 4.3.1. More detailed results, including the singular initial condition and 3d harmonic oscillator setting, are given in Appendix C.4.3.

4.3.3 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 4.2 (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 $\int_{\mathbb{R}^d} |\psi(x, t)|^2 \mathrm{d}x$ 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 C.4.5). 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 and Heyl, 2020] did not yield satisfactory results for any d. Additional details are provided in Appendix C.3.2.

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 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 4.3 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.

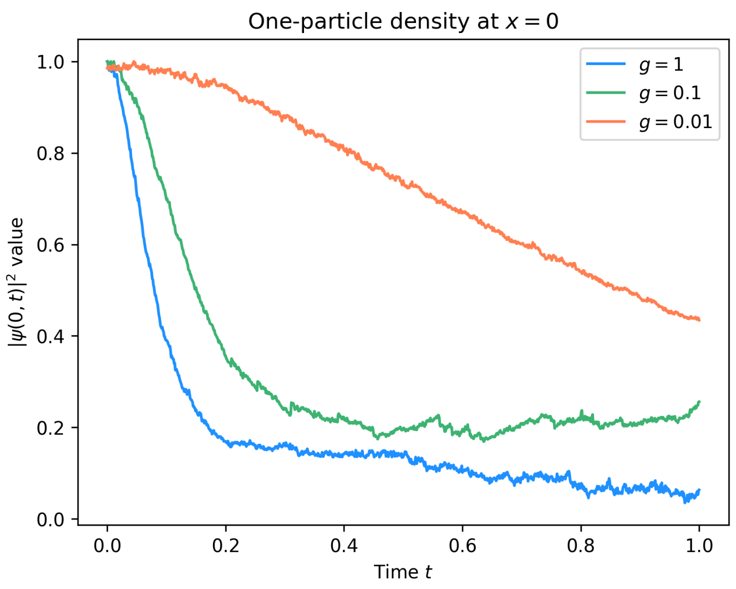


Figure 4.3: One-particle density of a system of 100 interacting bosons for varying interaction strength g. For a weaker interaction, the one-particle density is higher, indicating a more stable particle configuration. Conversely, for a stronger interaction, this value decreases, suggesting a more dispersed particle behavior.

4.3.4 Computational and memory complexity

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 4.3.1. 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 4.4 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 4.4 demonstrates the total training time of our version versus the problem 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 C.4.4.

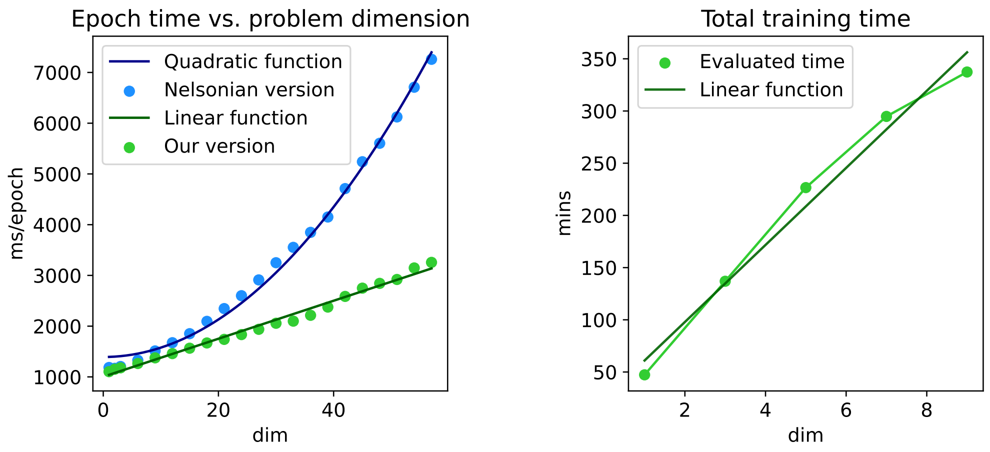


Figure 4.4: Empirical complexity evaluation of our method for the non-interacting system.

Interacting system We study the scaling capabilities of our DSM approach in the setting from Section 4.3.3, comparing the performance of our algorithm with a numerical solver based on the Crank–Nicolson method. Table 4.4 shows training time, time per epoch, and memory usage for our method. Table 4.3 reports time and memory usage of the Crank–Nicolson method solver. More details and illustrations of obtained solutions are given in Appendix C.4.5.

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 4.3: 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 4.4: 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

4.4 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.

4.5 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 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 a discussion of future work in Appendix C.9.

CHAPTER 5

FERMIONIC DEEP STOCHASTIC MECHANICS

5.1 Introduction

Partial differential equations (PDEs) in high dimensions form the mathematical foundation for modeling numerous natural phenomena. Given access to precise solutions of the time-dependent Schrödinger equation (TDSE) describing the interactions of many electrons in a molecule, a vast body of scientific problems could be addressed, including quantum chemistry [Cances et al., 2003, Nakatsuji, 2012], drug discovery [Ganesan et al., 2017, Heifetz, 2020], condensed matter physics [Boghosian and Taylor IV, 1998, Liu et al., 2013] and quantum computing [Grover, 2001, Papageorgiou and 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: classical grid-based methods exhibit exponential computational complexity with respect to system dimensionality [Bellman, 2010]. Traditional quantum chemistry methods for *time-independent* or ground state simulations, such as Hartree-Fock (HF) [Hartree, 1928] or Coupled Cluster [Čížek, 1966, 1969], exhibit prohibitive computational scaling of $O(n^7)$ with a system size n (the number of electrons), limiting their application to larger systems. Deep learning is a promising approach to circumvent this limitation [Poggio et al., 2017, Madala et al., 2023]. For example, recent deep learning approaches achieve a computational scaling of $O(n^k)$ where $2 \leq k \leq 4$ [Schütt et al., 2019, Pfau et al., 2020a, Keith et al., 2021], while sometimes exceeding the precision of the Coupled Cluster (CC) and Density Functional Theory [Kohn and Sham, 1996] methods. These approaches are focused primarily on the ground state problems with some extensions to quenched time dynamics and excited states. However, reliable time-dynamical simulations for larger systems without assuming additional constraints, such as quenched ground states, remain largely unexplored.

While generic deep learning approaches can be applied to solving TDSE [E and Yu, 2017, Han et al., 2018, Raissi et al., 2019, Weinan et al., 2021], this chapter shows that significant performance improvements can be achieved by developing an approach specific to fermionic TDSE by incorporating quantum physical structure into the deep learning algorithm itself.

A recent deep stochastic mechanics (DSM) framework [Orlova et al., 2023], described in Section 4.2, demonstrates promising results in terms of scaling with the system size for bosonic systems. However, its direct application to fermionic systems is limited due to the inherent antisymmetric nature of fermions, currently limiting DSM's impact to quantum chemistry research. In this work, we develop a fermionic extension of DSM (F-DSM) that addresses these limitations. This extension presents significant challenges beyond those encountered in bosonic systems, particularly in maintaining numerical stability while encoding antisymmetry. Although individual components of the solution exist, their integration demands new approaches to numerical stability, optimization procedures, and integration schemes.

In this work, we establish a foundation for time-dynamical simulations of molecular systems. We extend the existing DSM framework to fermions by developing a neural architecture that natively represents fermionic antisymmetry, and utilize a stochastic interpretation of quantum mechanics [Nelson, 2005] to effectively sample from a stochastic differential equation (SDE).

5.1.1 Problem formulation

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

$$i\hbar\partial_t \Psi(x,t) = \mathcal{H}\Psi(x,t), \tag{5.1}$$

$$\Psi(x,0) = \Psi_0(x), \tag{5.2}$$

where $\Psi : \mathcal{M} \times [0, T] \rightarrow \mathbb{C}$ represents the complex-valued wave function defined over a manifold $\mathcal{M}$. We consider fermionic systems where $\mathcal{M} = (\mathbb{R}^3 \times \pm 1)^n$ with ± 1 corresponding to spin states (up and down), and n denoting the number of particles. $\mathcal{H}$ is a self-adjoint operator acting on a Hilbert space of wave functions. Within the Born-Oppenheimer approximation [Born and Heisenberg, 1985], the Hamiltonian takes the form $\mathcal{H} = -\frac{\hbar^2}{2}\mathrm{Tr}(m^{-1}\nabla^2) + V(x)$, where $\nabla^2 = [\frac{\partial^2}{\partial x_i x_j}]_{i=1,j=1}^{n,n}$ is a Hessian operator, and $V(x)$ is a potential function. We use atomic units throughout this work, setting $\hbar = 1$ and $m = 1$. The probability density of finding a particle at position x at time t is $|\Psi(x,t)|^2$.

Molecular systems Our focus is on molecular systems with n_{el} electrons, and the potential $V(x)$ represents the Coulomb interactions. The many-body wave function $\Psi(x, t|R, Z, S)$ depends on the nuclei coordinates $R = \{R_I\}_{I=1}^{n_{\mathrm{nuc}}}$ and atomic numbers $Z = \{Z_I\}_{I=1}^{n_{\mathrm{nuc}}}$, where $R_I \in \mathbb{R}^3$ and $Z_I \in \mathbb{Z}^+$, and electronic spin configurations $S = \{S_i\}_{i=1}^{n_{\mathrm{el}}}$, where $S_i \in \pm 1$. In our notation here and below, we separate the particle coordinates $x_i \in \mathbb{R}^3$, where $i = 1, ..., n_{\mathrm{el}}$, and their corresponding spins for convenience. Not all functions are valid wave functions – particles must be indistinguishable, meaning $|\Psi|^2$ should be invariant to permutations of the particles. Additionally, the Pauli exclusion principle states that the probability of observing any two electrons in the same state must be zero. This is enforced by requiring the wave function for electronic systems to be antisymmetric. For any permutation $\pi \in \mathrm{Perm}_{n_{\mathrm{el}}}$, where $\mathrm{Perm}_{n_{\mathrm{el}}} : \{\pi : \{1, \ldots, n_{\mathrm{el}}\} \rightarrow \{1, \ldots, n_{\mathrm{el}}\}\}$ is a bijection from a set $\mathrm{Perm}(n_{\mathrm{el}})$ to itself, we require

$$\Psi(x_1, ..., x_{n_{\mathrm{el}}}, t|R, S, Z) = \sigma(\pi)\Psi(x_{\pi(1)}, ..., x_{\pi(n_{\mathrm{el}})}, t|R, S_{\pi(1)}, ..., S_{\pi(n_{\mathrm{el}})}, Z), \tag{5.3}$$

where $\sigma(\pi) = \pm 1$ is the parity or sign of permutation π. Given initial conditions in the form of samples drawn from the density $|\psi_0(x|R, Z, S)|^2$, we wish to draw samples $x \in \mathbb{R}^{3^n}$

from $|\psi(x,t|R,Z,S)|^2$ 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 $\Psi(x,t|R,Z,S)$ and subsequently sampling from the corresponding density, we directly sample from an approximation of $|\Psi(x,t|R,Z,S)|^2$, concentrating computation in high-density regions. In addition, this approach has the potential to generalize across varying molecular configurations R, Z, S. When regions where the density $|\Psi(x,t|R,Z,S)|^2$ lie in a latent low-dimensional space, our sampling strategy concentrates computation in that space, leading to the favorable scaling properties of our approach. Throughout this manuscript, we can use the abbreviated notation $\Psi(x,t)$ instead of $\Psi(x,t|R,Z,S)$ for simplicity.

5.1.2 Related work

Time-independent past works In quantum chemistry, substantial progress is made in the study of ground-state properties, leading to a variety of computational methods. The cornerstone of these methods is the Hartree-Fock approximation [Hartree, 1928], which treats the many-electron wave function as a single Slater determinant of one-electron orbitals. While HF captures about 99% of the total energy through mean-field electron-electron interactions, it neglects electron correlation, a crucial aspect of many systems. To capture electron correlation, several post-HF methods are developed. Configuration Interaction (CI) [Shavitt, 1977] systematically improves upon HF by including excited Slater determinants. Full Configuration Interaction (FullCI) [Knowles and Handy, 1984] provides exact solutions within a given basis set by including all possible electron excitations, but its factorial scaling makes it feasible only for small systems. Density Functional Theory (DFT) offers an alternative approach by working with electron density rather than the many-electron wave function. It typically scales as $O(n^4)$. While exact in principle, practical DFT relies on approximate exchange-correlation functionals.

An alternative perspective recognizes that experimental observations are limited to samples from the quantum density. This insight suggests focusing on density sampling rather than directly solving the Schrödinger equation, enabling system behavior prediction without physical experimentation. Variational Monte Carlo (VMC) methods estimate observable (energy) expectations rather than the wave function itself, achieving improved computational efficiency. In VMC, the wave function representation or ansatz typically combines a determinantal part (often from HF or DFT calculations) with a Jastrow factor that explicitly includes electron-electron correlations. The Jastrow factor can efficiently capture electron correlation effects that would require many determinants in traditional CI approaches. However, VMC results are inherently limited by the quality of the wave function representation or ansatz, and achieving high accuracy requires careful wave function optimization. Recent advances in variational wave function ansatz, particularly the introduction of neural quantum states (NQS) with variational Monte Carlo [Carleo and Troyer, 2017], produce remarkably accurate solutions to the electronic time-independent Schrödinger equation. NQS offers high-accuracy wave functions that capture system entanglement at a manageable computational cost [Hermann et al., 2020].

Time-dependent past works While significant progress is made in solving the stationary Schrödinger equation, its time-dependent counterpart presents greater challenges. This increased complexity stems from the quantum system's tendency to explore a much larger portion of the Hilbert space compared to ground-state problems. Therefore, heuristics developed for time-independent problems are not directly applicable. A common strategy for approximating time-dependent dynamics is to neglect electron–electron correlations entirely, as in mean-field theories. The seminal time-dependent Hartree-Fock (TDHF) framework introduced by Dirac and Frenkel [Dirac, 1930, Frenkel, 1934] follows this approach. While TDHF performs reasonably well for weakly correlated systems, such as closed-shell molecules under small perturbations, it fails in the presence of strong correlation. Efforts to go beyond the mean-field level led to the development of multi-configuration time-dependent Hartree-Fock (MCTDHF) [Li et al., 2005, Lode et al., 2020], which extends the ansatz to include multiple electronic configurations. However, these methods suffer from exponential scaling with system size and remain computationally impractical beyond small systems. Real-time time-dependent density functional theory (TDDFT) [Yabana and Bertsch, 1996, Ullrich, 2011] is a popular method, particularly in condensed-phase simulations and optical response studies. However, TDDFT's reliance on adiabatic and local approximations in exchange-correlation functionals limits its accuracy for strongly correlated dynamics. Other formulations, such as time-dependent multi-configuration self-consistent field (TD-MCSCF) [Sato and Ishikawa, 2013, Liu et al., 2019], configuration interaction (CI) [Krause et al., 2005, Schlegel et al., 2007, Krause et al., 2007, DePrince III et al., 2011, Woźniak et al., 2022], coupled-cluster (CC) [Luppi and Head-Gordon, 2012, Nascimento and DePrince III, 2016, Skeidsvoll et al., 2020], and tensor network approaches [Haegeman et al., 2011, Pichler et al., 2016], offer higher expressivity but are either limited to short time scales or small systems due to prohibitive scaling. Time-evolving density matrix renormalization group (t-DMRG) methods [Cazalilla and Marston, 2002] were successfully applied in 1D lattice models but are not applicable to real-space molecular fermionic dynamics.

As for variational methods, there are works that explore time-dependent Variational Monte Carlo (t-VMC) schemes [Carleo and Troyer, 2017, Yao et al., 2021, Nys et al., 2024] for simulating many-body quantum systems, including fermionic ones. Those approaches suffer from an accumulation of errors coming from the sequential propagation of the small time-step dynamics, where each step depends on the accuracy of previous steps. While our F-DSM approach also relies on wave function representation quality, we address several shortcomings of t-VMC methods. Our global time parameterization optimizes the entire trajectory simultaneously against the TDSE, avoiding the error accumulation inherent to sequential optimization schemes. Our stochastic reformulation circumvents the expensive

MCMC sampling and matrix inversion steps in optimization that lead to instability in t-VMC methods. The original DSM paper [Orlova et al., 2023] provides experimental evidence of these instability challenges. 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. Furthermore, there is not a single t-VMC method that is shown to work beyond quenched ground states of electronic systems. In summary, while many theoretically appealing approaches exist, there is currently no widely adopted, scalable, and robust framework for the real-time simulation of strongly correlated many-electron systems.

5.2 Contributions

Our work is based on the DSM approach [Orlova et al., 2023] that uses a stochastic interpretation of quantum mechanics by Nelson [Nelson, 1966a]. We extend the DSM framework to molecular systems. We highlight the main contributions of this work as follows:

- Paradigm-shifting ansatz architecture: We introduce a novel ansatz that is fundamentally different from conventional Slater-Jastrow-Backflow (SJB) architectures. Unlike SJB models, which express the wave function as a linear combination of antisymmetric basis functions, our approach learns a nonlinear composition of antisymmetric components via a stack of sign-equivariant neural layers. This nonlinear recombination enhances the network's ability to capture rich many-body correlations beyond the expressivity of additive determinant expansions.

- Orbital positional encoding (OPE): Another key aspect of our architecture design is a new positional encoding methodology that represents electronic orbitals through learnable periodic functions parameterized via the fundamental Clifford group [Clifford, 1873, 1882]. Clifford algebra provides a unified representation of spatial position, orientation, and spin, making it particularly well-suited for modeling structure of many-electron systems. A key property of our OPE design is that it is *parameter-efficient: the number of learnable parameters is independent of the number of electrons, orbitals, or nuclei, enabling scalability to larger systems without increasing model size*. In addition, this approach enables potential extrapolation to atomic numbers beyond the training distribution, a capability previously observed only in specialized force field modeling contexts [Plé et al., 2023]. In the following layers, rather than implementing hard-coded equivariance constraints, our architecture incorporates symmetry through specialized layers comprising low-rank decompositions with equivariant factors. Our full ansatz includes antisymmetric and symmetric branches, sign-equivariant mixing layers that preserve antisymmetry under particle exchange, and explicit cusp conditions.

- Stochastic mechanics framework for time dynamics simulation: By leveraging Nelson's stochastization formalism, we establish a global time variational method that circumvents the dimensionally cursed rejection sampling required in traditional VMC approaches, enabling efficient time-dependent simulation. By pretraining our ansatz on a conventional quantum chemistry method, we observe a notable phenomenon: our approach effectively projects these predictions into a physically correct solution space. This suggests that our method not only addresses the methodological limitations of traditional approaches but can actually refine and improve existing high-level quantum chemistry calculations by enforcing proper physical constraints.

- Experiments: We validate our approach through experiments on a hydrogen molecule system, demonstrating its ability to handle challenging initial conditions. Notably, even when starting from an excited state, a scenario that typically presents significant computational difficulties, our method successfully simulates reasonable quantum dynamics. This capability shows the robustness of our approach in modeling quantum systems beyond ground state configurations.

5.3 Extending DSM framework to molecular systems: F-DSM

We present a novel neural network architecture designed to represent time-dependent many-electron wave functions while preserving essential physical symmetries. Our architecture combines orbital features with sign-equivariant mixing layers and explicit cusp conditions to achieve proper fermionic behavior. We start from a general overview of the DSM framework.

5.3.1 General DSM framework

The DSM framework builds upon Nelson's stochastic interpretation of quantum mechanics [Nelson, 1966a], which establishes an equivalence between the Schrödinger equation and a diffusion process. This equivalence allows quantum mechanics to be formulated in terms of SDEs, providing an alternative computational approach to quantum simulations.

For a given wave function in polar coordinates $\Psi(x,t) = \sqrt{\rho(x,t)}e^{iS(x,t)}$, we define two velocity fields:

$$\begin{aligned} v(x,t) &= \mathfrak{Re}\nabla \log \Psi(x,t) = \nabla S(x,t), \\ u(x,t) &= \mathfrak{Im}\nabla \log \Psi(x,t) = \frac{1}{2}\nabla \log \rho(x,t). \end{aligned} \tag{5.4}$$

Let W_t denote a time-symmetric reciprocal Wiener process. For any [finite-energy] solution of TDSE (5.1) with an initial condition $\Psi(\cdot,0) = \Psi_0$, we have the following stochastic process

$$\begin{aligned} \mathrm{d}Y(t) &= (v(Y(t),t) + u(Y(t),t))\mathrm{d}t + \mathrm{d}W_t, \\ Y(0) &\sim \left|\Psi_0\right|^2. \end{aligned} \tag{5.5}$$

This Nelsonian process (5.5) corresponds to sampling from $\rho = |\Psi(x,t)|^2$, or $Y(t) \sim |\Psi(x,t)|^2$. [1] Any numerical integrator can be used to obtain samples from the diffusion

1. Theoretical results by Kuipers [2023] establish that this formulation remains valid even in the case of multi-valued Ψ, which might be the case for fermions. This mathematical foundation justifies our sam-

process. We primarily use adaptive Runge-Kutta methods to maintain numerical stability throughout the integration process.

DSM approach approximates the Nelsonian process $Y(t)$ with a process $X(t)$ where the wave function is parametrized via a neural network with parameters θ:

$$
\begin{aligned}
\mathrm{d}X(t) &= (\mathfrak{Re}\nabla \log \Psi_\theta(X(t),t) + \mathfrak{Im}\nabla \log \Psi_\theta(X(t),t))\mathrm{d}t + \mathrm{d}W \\
&= (v_\theta(X(t),t) + u_\theta(X(t),t))\mathrm{d}t + \mathrm{d}W, \qquad (5.6) \\
X(0) &\sim \left|\Psi_0\right|^2.
\end{aligned}
$$

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

$$
X_{i+1} = X_i + (v_\theta(X_i,t_i) + \nu u_\theta(X_i,t_i))\epsilon + \mathcal{N}\big(0, \frac{\nu\hbar}{m}\epsilon I_d\big), \qquad (5.7)
$$

where $\epsilon > 0$ denotes a step size, $0 \le i < \frac{T}{\epsilon}$, and $\mathcal{N}(0, I_d)$ is a Gaussian distribution. All these details can be found in Section 4.2.

5.3.2 F-DSM approach

Unlike the original DSM framework, which directly employs neural networks to parametrize the velocities u and v (effectively modeling gradients of the logarithm of the wave function), our approach for fermionic systems requires a fundamentally different parametrization strategy. To properly encode the antisymmetric properties of fermions, we *directly parameterize the wave function itself rather than its derivatives*. Specifically, we implement a neural network ansatz Ψ_θ that outputs wave function values (i.e., Ψ_θ inputs location x and time t in addition to nuclei R, spins S and atomic numbers Z, and outputs $\Psi_\theta(x,t|R,S,Z)$), from which we subsequently compute gradients to use in Equation (5.6). This methodological dis-

pling methodology, ensuring that the stochastic trajectories correctly sample from the quantum probability distribution at each time step.

tinction enables our framework to capture the sign-changing behavior required by the Pauli exclusion principle while maintaining the computational advantages of the diffusion-based sampling approach.

To learn this process in practice, the following loss functions are defined by substituting $\Psi(x,t) = \sqrt{\rho(x,t)}e^{iS(x,t)}$ in polar coordinates into the Schrödinger equation (for the detailed derivation, we refer to Orlova et al. [2023]):

$$L_1(v_\theta, u_\theta) = \int_0^T \mathbb{E}^X \left\| \partial_t u_\theta(X(t),t) - \mathcal{D}_u[v_\theta, u_\theta, X(t), t] \right\|^2 \mathrm{d}t, \tag{5.8}$$

$$L_2(v_\theta, u_\theta) = \int_0^T \mathbb{E}^X \left\| \partial_t v_\theta(X(t),t) - \mathcal{D}_v[v_\theta, u_\theta, X(t), t] \right\|^2 \mathrm{d}t, \tag{5.9}$$

$$L_3(v_\theta, u_\theta) = \mathbb{E}^X \| u_\theta(X(0),0) - u_0(X(0)) \|^2, \tag{5.10}$$

$$L_4(v_\theta, u_\theta) = \mathbb{E}^X \| v_\theta(X(0),0) - v_0(X(0)) \|^2, \tag{5.11}$$

where $u_0 = \nabla S(x,0)$, $v_0 = \frac{1}{2}\nabla \log \rho(x,0)$, and

$$\mathcal{D}_u[v,u,x,t] = -\nabla\langle v(x,t), u(x,t)\rangle - \frac{1}{2}\nabla\langle \nabla, v(x,t)\rangle, \tag{5.12}$$

$$\mathcal{D}_v[v,u,x,t] = -\frac{1}{m}\nabla V(x,t) + \frac{1}{2}\nabla\|u(x,t)\|^2 - \frac{1}{2}\nabla\|v(x,t)\|^2 + \frac{1}{2}\nabla\langle \nabla, u(x,t)\rangle. \tag{5.13}$$

Finally, we define a combined loss using a weighted sum with $w_i > 0$:

$$\mathcal{L}(\theta) = \sum_{i=1}^{4} w_i L_i(v_\theta, u_\theta). \tag{5.14}$$

The basic idea of our approach is to sample new trajectories using the integrated Equation (5.6) for each epoch. These trajectories are then used to compute stochastic estimates of the loss from Equation (5.14), and then we backpropagate gradients of the loss to update the parameters θ. During inference, the integrated version of Equation (4.5) (for example, Equation (5.7)) can be used to produce samples.

This sampling procedure offers significant computational advantages for simulating TDSE compared to traditional variational methods. As established in computational physics [Gelman et al., 1997, Beskos et al., 2013], the MCMC sampling computational overhead scales unfavorably with system dimensionality, often exponentially, due to the rapidly increasing number of burn-in steps needed for proper convergence. By avoiding repeated MCMC sampling throughout the temporal evolution, we enable more efficient computation, assuming the ability to sample from the initial wave function Ψ_0. Specifically, methods relying on MCMC sampling require M distinct MCMC procedures to compute observables at M temporal points, while our approach requires a single MCMC application for the initial state, irrespective of the number of evaluation points.

Two-phase training approach for fermionic systems A critical consideration in applying the F-DSM framework to fermionic systems involves addressing the inherent mathematical pathologies of electronic wave functions within the Born-Oppenheimer approximation. These wave functions exhibit non-analyticity manifested through cusps that can break numerical stability during SDE integration, while maintaining mathematical properties essential for stable dynamics.

Dynamic electron correlation arises from the physical Coulomb repulsion between electrons. When electrons coincide spatially, the electronic molecular Hamiltonian exhibits singular behavior, while the local energy $\mathcal{H}\Psi/\Psi$ remains constant. To compensate for this singularity of $\mathcal{H}$, the wave function Ψ must have a nondifferentiable behavior

$$\lim_{x_i \to x_j} \frac{\partial \Psi}{\partial x}\Big|_{x_i = x_j} = \frac{1}{2} Z \Psi(x_i = x_j), \qquad (5.15)$$

at electron coalescence points $x_i = x_j$ resembling a cusp, thus called electronic cusp condition or Kato's cusp condition [Kato, 1957]. This discontinuity exactly cancels the Coulomb singularity in the potential energy, preventing the kinetic energy from becoming infinite. Without properly satisfying this cusp condition, the kinetic energy contribution would diverge, leading to infinite total energy. The same holds for electron-nuclei interactions: the wave function's derivative has a discontinuity at the position where an electron coincides with a nucleus.

In the context of ground-state problems, conventional wave function $\psi(x)$ approximations using single particle orbital products, such as Slater-type orbitals [Slater, 1930], Gaussian-type orbitals [Hehre et al., 1969] or correlation consistent basis sets by Dunning [Dunning Jr, 1989], struggle to capture the nondifferentiable characteristics of the exact wave function. This limitation necessitates extensive basis set expansions and contributes to the slow convergence of basis set extrapolation methods, particularly when modeling the dynamic behavior of electrons in close proximity. While methods like FullCI or HF can achieve impressive energy accuracy, they might fail to satisfy these cusp conditions rigorously. When such wave functions serve as initial conditions for our SDE-based approach, they can introduce gradient instabilities that accumulate during integration, potentially rendering the entire simulation invalid.

So, stochastic optimization approaches for the TDSE exhibit sensitivity to violations of mathematical constraints compared to linear projection methods operating on predefined basis sets. Consequently, conventional calculations from methods like FullCI might not be suitable initial conditions Ψ_0 without modification, regardless of their energy recovery precision.

To address this challenge, we propose the following two-phase training scheme:

1. **IC pretraining** Firstly, we train our ansatz (architecture detailed in Section 5.4) to accurately predict initial conditions at $t = 0$, denoted as $\Psi_\theta^{\mathrm{IC}}$. The training can leverage reference data from established quantum chemistry methods such as FullCI or HF. Our ansatz incorporates explicit representations for fermionic behavior and electron cusps, ensuring proper handling of wave function singularities. We use an enhanced initial condition loss function that builds upon Equations (5.10) and (5.11) by adding derivative terms to capture all singularities:

$$\begin{aligned} L_{u_0}(v_\theta, u_\theta) =& \mathbb{E}^X \|u_\theta(X(0), 0) - u_0(X(0))\| \\ &+ \mathbb{E}^X[1 + \log(\|\nabla u_\theta(X(0), 0) - \nabla u_0(X(0))\|^2)] \\ &+ \mathbb{E}^X[1 + \log(\|\nabla^2 u_\theta(X(0), 0) - \nabla^2 u_0(X(0))\|^2)], \end{aligned} \tag{5.16}$$

$$\begin{aligned} L_{v_0}(v_\theta, u_\theta) =& \mathbb{E}^X \|v_\theta(X(0), 0) - v_0(X(0))\| \\ &+ \mathbb{E}^X[1 + \log(\|\nabla v_\theta(X(0), 0) - \nabla v_0(X(0))\|^2)] \\ &+ \mathbb{E}^X[1 + \log(\|\nabla^2 v_\theta(X(0), 0) - \nabla^2 v_0(X(0))\|^2, \end{aligned} \tag{5.17}$$

While this pretraining step can be bypassed by initializing with the conventional quantum chemistry methods, our experiments demonstrate that such approaches lead to significantly degraded performance (Section 5.5.2). Therefore, we use our ansatz architecture, detailed in Section 5.4, with $t = 0$ for this step. The framework nevertheless accommodates alternative methods or representations capable of effectively simulating antisymmetric wave functions.

2. **Regular DSM training** The second phase is based on our standard DSM framework with an important modification: rather than deriving initial conditions directly from the conventional quantum chemistry methods, we utilize the pretrained model $\Psi_\theta^{\mathrm{IC}}$ as the source of these conditions. Here, we train our time-dependent ansatz $\Psi_\theta(x, t)$ detailed in the next section, to learn the full wave function evolution. In our

experiments, we use identical architectures for both the pretraining and DSM training phases; however, the weights of $\Psi_\theta^{\mathrm{IC}}$ are frozen during this stage to preserve the learned initial condition representations. We use the standard loss function defined in Equation (5.14).

We observe that this scheme enables accurate simulation of fermionic many-body dynamics without the numerical instabilities that would otherwise arise from singularities in the electronic wave function. The pretraining methodology plays a critical role in our overall framework, ensuring that the subsequent F-DSM optimization proceeds from a physically correct and numerically stable initial state of a fermionic system. Our experiments with the hydrogen molecule demonstrate that neural networks trained with this approach recover the reasonable quantum dynamics.

5.4 Ansatz representation

The form of the wave function Ψ must be restricted to antisymmetric functions to avoid collapsing onto non-physical solutions. In the context of ground state problems, this is most commonly done by taking a set of one-electron orbitals $\phi_1, ..., \phi_{n_{\mathrm{orb}}}$ and approximating the wave function as a linear combination of antisymmetrized tensor products, called Slater determinants, of those functions

$$\sum_\pi (-1)^\pi \prod_i \phi_i^k(\pi(x)) = \begin{vmatrix} \phi_i^k(x_1) & \cdots & \phi_i^k(x_{n_{\mathrm{el}}}) \\ \vdots & \ddots & \vdots \\ \phi_m^k(x_1) & \cdots & \phi_m^k(x_{n_{\mathrm{el}}}) \end{vmatrix} = \det[\phi_i^k(x_j)] = \det[\Phi^k] \tag{5.18}$$

$$\psi(x_1, \ldots, x_{n_{\mathrm{el}}}) = \sum_k \omega_k \det[\Phi^k], \tag{5.19}$$

where $\{\phi_1^k, \ldots, \phi_m^k\}$ is a subset of m of the n_{orb} orbitals, the sum in Equation (5.18) is taken over all permutations π of the electron indices, and the sum in Equation (5.19) is over all subsets of m orbitals. The difficulty is that the number of possible Slater determinants rises exponentially with the system size, restricting methods like FullCI to tiny molecules, even with recent advances [Booth and Alavi, 2010]. In practice, QMC methods commonly adopt the Slater–Jastrow ansatz [Foulkes et al., 2001], which combines a truncated linear combination of Slater determinants with a multiplicative Jastrow factor to capture close-range correlations. The Jastrow factor is usually a symmetric term designed to capture electron–electron and electron–nucleus correlations, and constructed as a product of functions depending on pairwise and triplet distances. In addition, the backflow transformation [Feynman and Cohen, 1956] can be applied; it shifts each electron's position based on nearby electron configurations before evaluating the orbitals. FermiNet [Pfau et al., 2020a] represents each orbital as a learnable function of all electron coordinates and spins that are then used to construct Slater determinants, with multiple such determinants enhancing expressivity. The expressivity of these methods remains limited due to their fundamentally linear combination structure and fixed orbital representations. Though there are other alternitives [Bajdich et al., 2006, von Glehn et al., 2022], the Slater–Jastrow–backflow form remains the standard for continuous-space many-electron systems.

We propose a novel way of representing a time-dependent ansatz as a neural network with parameters θ that nonlinearly composes symmetric and antisymmetric components. Due to the intrinsic complexity of the ansatz, we first present a general functional form, followed by a detailed breakdown. At the highest level, our time-dependent wave function ansatz can be expressed as:

$$\Psi_\theta(x,t|R,Z,S) = e^{J_\theta(x,R,Z,S)}\Psi_\theta^{(L_{\text{mix}})} = e^{J_\theta(x,R,Z,S)}\underbrace{f_\theta^{\text{mix}}\Big(f_\theta^{\text{mix}}(\dots}_{L_{\text{mix}}} f_\theta^{\text{init}}(x,t,R,Z,S))\Big), \tag{5.20}$$

where $e_\theta^J(x,R,Z,S)$ is a Jastrow factor with the cusp conditions encoded, and the term $\Psi_\theta^{(L_{\text{mix}})}$ denotes the output of our sign-equivariant mixing layer f_θ^{mix} that is stacked L_{mix} times. Beyond these primary elements, our architecture has other modules organized into an initial block f_θ^{init} that processes and transforms different aspects of the quantum system representation:

- Orbital positional encoding P_θ that leverages concepts from Clifford algebra to generate orbitals, whose parameterization does not depend on a maximal atomic number, nor on the number of electrons, nor on the number of atoms, yet is expressive enough. A more detailed description is given in Section 5.4.1

- Interaction towers $\Omega_\theta^{\text{e-o}}$, $\Omega_\theta^{\text{o-n}}$, $\Omega_\theta^{\text{e-n}}$ that process electron-orbital, and orbital-nucleus, electron-nucleus interactions via attention mechanisms that are fused with a special layer Γ_θ later. More details are provided in Section 5.4.2.

- Symmetric s_θ and antisymmetric a_θ embeddings for components with different symmetry properties; a time embedding $\tau_\theta(t)$. They are described in Section 5.4.4.

These modules process the model's inputs x, t, R, Z, S, generating inputs for the symmetric and antisymmetric embeddings used in the sign-equivariant mixer f_θ^{mix}, and ultimately the wavefunction itself. Figure 5.1 illustrates a schematic overview of our ansatz architecture. We provide a more detailed description of every module below.

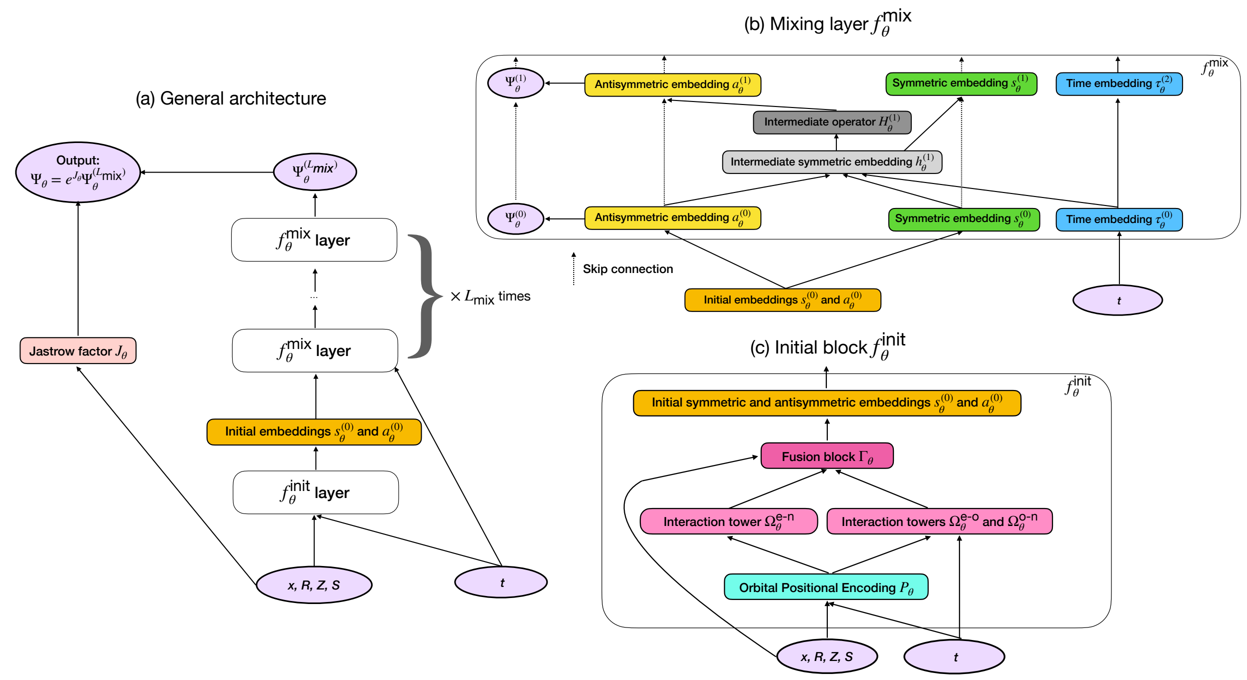

(a) General architecture
Output: $\Psi_\theta = e^{J_\theta}\Psi_\theta^{(L_{\text{mix}})}$
$\Psi_\theta^{(L_{mix})}$
f_θ^{mix} layer
f_θ^{mix} layer
$\times L_{\text{mix}}$ times
Jastrow factor J_θ
Initial embeddings $s_\theta^{(0)}$ and $a_\theta^{(0)}$
f_θ^{init} layer
x, R, Z, S
t
(b) Mixing layer f_θ^{mix}
f_θ^{mix}
$\Psi_\theta^{(1)}$
Antisymmetric embedding $a_\theta^{(1)}$
Symmetric embedding $s_\theta^{(1)}$
Time embedding $\tau_\theta^{(2)}$
Intermediate operator $H_\theta^{(1)}$
Intermediate symmetric embedding $h_\theta^{(1)}$
$\Psi_\theta^{(0)}$
Antisymmetric embedding $a_\theta^{(0)}$
Symmetric embedding $s_\theta^{(0)}$
Time embedding $\tau_\theta^{(0)}$
Skip connection
Initial embeddings $s_\theta^{(0)}$ and $a_\theta^{(0)}$
t
(c) Initial block f_θ^{init}
f_θ^{init}
Initial symmetric and antisymmetric embeddings $s_\theta^{(0)}$ and $a_\theta^{(0)}$
Fusion block Γ_θ
Interaction tower $\Omega_\theta^{\text{e-n}}$
Interaction towers $\Omega_\theta^{\text{e-o}}$ and $\Omega_\theta^{\text{o-n}}$
Orbital Positional Encoding P_θ
x, R, Z, S
t

Figure 5.1: Architecture of the time-dependent fermionic ansatz. (a) General architecture showing the initial block f_θ^{init} followed by L_{mix} sign-equivariant mixing layers f_θ^{mix}, combined with the Jastrow factor to produce the final wave function Ψ_θ. (b) Structure of the sign-equivariant mixing layer f_θ^{mix}, which processes symmetric, antisymmetric, and time embeddings. (c) Initial block f_θ^{init}, incorporating orbital positional encoding, interaction towers, and fusion block to generate initial embeddings $s_\theta^{(0)}, a_\theta^{(0)}$ from the input configuration (x, t, R, Z, S).

5.4.1 Orbital positional encoding (OPE)

We introduce a learnable orbital positional encoding scheme based on Clifford algebra. Unlike traditional approaches that assign a fixed set of orbitals to each electron or rely on externally computed orbitals (e.g., from Hartree–Fock), our method learns orbitals end-to-end.

Let us briefly introduce some concept from Clifford algebra. We work with the Clifford algebra $\mathfrak{cl}_{(3,0)}(\mathbb{R})$, which is defined as the factor algebra of the free algebra $\mathbb{R}[e_1, e_2, e_3]$ over ideal generated by polynomials $e_i^2 - \mathfrak{1}$, $i = 1, 2, 3$. Its linear basis can be chosen as $\{\mathfrak{1}, e_1, e_2, e_3, e_1e_2e_3, e_2e_3, e_1e_3, e_1e_2\}$. By denoting $e_1e_2e_3 =: \mathfrak{i}, \mathfrak{i}^2 = -\mathfrak{1}$, it can be rewritten as $\{1, e_1, e_2, e_3, \mathfrak{i}, \mathfrak{i}e_1, \mathfrak{i}e_2, \mathfrak{i}e_3\}$. Under parity transformations, $\mathfrak{1}$ is even (does not change upon reflection of the basis) and $\mathfrak{i}$ is odd. So, $\mathfrak{i}$ behaves analogously to the determinant operator on the basis e_1, e_2, e_3.

We can interpret this space as $\mathbb{R}^{2\times 4}$: $\mathfrak{cl}_{(3,0)} \simeq \mathbb{R}^{2\times 4}$, where the first index tracks parity (presence of $\mathfrak{i}$) and the second spans the subspace $\{1, e_1, e_2, e_3\}$. Moreover, the span of $\{\mathfrak{1}, \mathfrak{i}e_1, \mathfrak{i}e_2, \mathfrak{i}e_3\} \simeq \mathbb{H}$ – quaternionic algebra, providing a quaternionic representation of SO(3) and facilitating efficient computation of rotations and spin transformations.

This is leveraged in both encoding spatial structure and spin, and allows to computationally simplify implementation of a multiplication table. For more details on mathematical foundations of Clifford algebras, we refer readers to Hitzer [2012], Simeon [2019].

We are mostly interested in a product xy in the Clifford algebra, which we generalize over tensors of shape $\mathbb{R}^{2\times 4\times d_{\text{hid}}}$ to support batched computation (multiplying many elements at once). We define the power operation $\text{pow}(x, n)$ recursively by: $\text{pow}(x, n) = x\text{pow}(x, n-1)$, $\text{pow}(x, 0) = \mathbb{1}$. We normalize it for stability as $\text{pow}(x, n) = \frac{x^n}{\|x^n\|}$. Due to the presence of divisors of zero in the Clifford algebra (i.e., pairs of non-zero elements $xy = \mathfrak{o}$), we impose a constraint on learnable encoding parameters $\theta \in \mathbb{R}^{2\times 4\times d_{\text{hid}}}$: $\sum_j \prod_i \theta_{ij} = 0$, $\|\theta\|_{\mathbb{R}^{2\times 4}} = 1$. This ensures that $\text{pow}(x, n) \neq 0$ for all n and avoids norm blow-up.

The central motivation for our positional encoding scheme is to construct expressive orbital features from a small set of learnable Clifford elements. Such parametrization includes all elements of the fundamental Clifford group, which includes $O(4)$ as a subgroup. Given elements $x, y \in \mathfrak{cl}3, 0(\mathbb{R})$ it holds almost surely (in the measure-theoretic sense) that $\{x^0 y, x^1 y, x^2 y, x^3 y\}$ is linearly independent. If y encodes electron positions, then one learnable x is enough to be able to generate 4×4 non-degenerate determinant (otherwise, orbitals are not really distinct, and the wave function becomes zero everywhere). To make it more expressive, we repeat this $d_{hid} > 1$ times, which should allow us to handle arbitrary large systems, while being able to produce non-zero determinants, which are important to represent the electronic antisymmetry.

We define a shared orbital positional encoding function $\mathfrak{pe}_\theta(x, t, R, S, Z) \in \mathbb{R}^{n_{\text{el}}\times n_{\text{orb}}\times n_{\text{nuc}}\times 2\times 4\times d_{\text{hid}}}$

that produces embedding features for every electron–orbital–nucleus triplet. We set $n_{\text{orb}} = n_{\text{el}}$ but n_{orb} can be arbitrarily specified. The OPE tensor P has elements indexed as (i, j, k, p, c, s), where i identifies an electron, j an orbital, and k a nucleus. The remaining indices correspond to the feature space: $p \in 0, 1$ represents parity, $c \in \{0, 1, 2, 3\}$ denotes channels, and s indexes the hidden dimension. Each element of P is defined as:

$$\begin{aligned} P_{ijkpcs} = \mathfrak{pe}_\theta(x, t, R, S, Z)_{ijk} =& \text{pow}(\theta^{\text{pos-emb}}(t), j)\text{pow}(\theta^{\text{atomic-number-emb}}(t), Z_k - 1) \\ & \times \text{EmbedClifford}(x_i - R_k, \theta^{\text{spin-emb}}_{S_i}(t)) \end{aligned} \tag{5.21}$$

where $\text{pow}(x, n)$ is the repeated Clifford algebra multiplication defined above, $\theta^{\text{atomic-number-emb}}$ is an embedding representing all atoms, $\theta^{\text{pos-emb}}$ is a representation for all orbitals, and $\theta^{spin}_{S_i}$ is an embedding for the spin label. These learnable parameters $\theta \in \mathbb{R}^{2\times4\times d_{\text{hid}}}$ are interpreted as unit-norm elements of the Clifford algebra $\mathfrak{cl}_{3,0}(\mathbb{R})$, with additional constraints to ensure nondegeneracy as discussed above. EmbedClifford is an operation that embeds real space and quaternionic spin representations into the Clifford algebra: a homomorphism $\mathcal{F} : \mathbb{H} \times \mathbb{R}^3 \to \mathfrak{cl}_{(3,0)}(\mathbb{R})$, where $\mathbb{H}$ denotes the quaternion algebra and $\mathfrak{cl}_{(3,0)}(\mathbb{R})$ is the Clifford algebra.

Importantly, the atomic-number-dependent term in our OPE induces a periodic structure in the embedding space, analogous to sinusoidal encodings, but generalized via Clifford algebra. It can also be conceptualized as a parametrization of the periodic table. Since Clifford powers encompass both periodic and exponential-like behavior (via their matrix representations), the resulting encodings offer a richer function class than traditional trigonometric positional embeddings used in attention-based models [Vaswani et al., 2017]. In particular, this allows the model to express both periodic and non-periodic dependencies, while preserving equivariance under rotation and reflection through the algebraic structure.

5.4.2 Interaction towers

To model different classes of physical interactions, we adopt a multi-tower design. Each tower targets a specific interaction channel: (1) $\Omega_\theta^{\text{e-o}}$ implements multi-head self-attention (MHSA) [Vaswani et al., 2017, von Glehn et al., 2022] over orbital axis, modeling how electrons distribute across and interact with available orbitals; (2) $\Omega_\theta^{\text{o-n}}$ utilizes a MHSA over the nuclei axis to model how orbitals are shaped and influenced by nuclear positions; (3) $\Omega_\theta^{\text{e-n}}$ employs geometric product operations to capture the interactions between electrons and nuclei, representing the Coulombic attractions that bind electrons to atomic centers. Each attention module operates on input embeddings P_{ijkpcs}, indexed over electron (i), orbital (j), nucleus (k), parity $(p \in 0, 1)$, channel $(c \in 1, 2, 3, 4)$, and hidden feature index (s).

After obtaining the positional encoding embedding P_{ijkpcs}, we compress it into a more tractable intermediate representation by aggregating hidden dimensions via linear layers and SiLU activations. It is done in the following way:

$$\Omega_{ijpcs}^{(0,\text{el, nuc})} = \sum_{s''} W_{spcs''}^{(0)} n_{\text{el}}^{-1} \sum_{j} (\text{SiLU}(\sum_{s'} W_{s''pcs'}^{(1)} P_{ijkpcs'} + \mathbb{1}_{p=1 \wedge c>1} b_{s''pc}) + \mathbb{1}_{p=1 \wedge c>1} b_{spc}, \tag{5.22}$$

$$\Omega_{ijpcs}^{(0,\text{el, orb})} = \sum_{s''} W_{spcs''}^{(0)} n_{\text{atoms}}^{-1} \sum_{k} (\text{SiLU}(\sum_{s'} W_{s''pcs'}^{(1)} P_{ijkpcs'} + \mathbb{1}_{p=1 \wedge c>1} b_{s''pc}) + \mathbb{1}_{p=1 \wedge c>1} b_{spc}, \tag{5.23}$$

$$\Omega_{ijpcs}^{(0,\text{orb, nuc})} = \sum_{s''} W_{spcs''}^{(0)} n_{\text{el}}^{-1} \sum_{i} (\text{SiLU}(\sum_{s'} W_{s''pcs'}^{(1)} P_{ijkpcs'} + \mathbb{1}_{p=1 \wedge c>1} b_{s''pc}) + \mathbb{1}_{p=1 \wedge c>1} b_{spc}, \tag{5.24}$$

which are our initial values for the towers. Each tower is updated over L ($L_{\text{el, orb}}$, $L_{\text{el, nuc}}$) layers using a standard residual + normalization + multi-head self-attention architecture

[Vaswani et al., 2017]. At layer $l+1$, the electron–orbital tower is updated as:

$$\begin{aligned}\Omega^{(l+1,\text{el, orb})}_{ijpcs} &= \text{LayerNorm}(\Omega^{(l,\text{el, orb})}_{ijpcs} \\ +\frac{1}{d_{\text{heads}}}\sum_h \text{MultiHeadSelfAttention}(&\sum_{s'} W^{(l+1,\text{el,nuc},*)}_{i,j,h,spcs'}\big(\theta^{(l+1,\text{el,orb})}_{s'}\big)^j \Omega^{(l,\text{el, orb})}_{ijpcs} \\ &\mid *\,(\in k,q,v))),\end{aligned} \tag{5.25}$$

where k,q,v are key, query, value, index, index h as a head index. The same holds for the orbital–nucleus tower.

The electron–nucleus tower employs a Clifford-aware geometric message passing scheme. Instead of attention, it aggregates information using weighted local interactions:

$$\begin{aligned}\Omega^{(l+1,\text{el, nuc})}_{ikpcs} = \text{LayerNorm}\Big[\Omega^{(l,\text{el, nuc})}_{ikpcs} &+ \sum_{s'}\sum_{s''} W^{(l+1,\text{geom})}_{ikss's''}\sum_{i'} \\ &\Bigg(\sum_{k'} \frac{e^{-\frac{\|x_i-x_j\|^2}{\sigma^2_{\text{el-el}}}}}{1+\sum_j e^{-\frac{\|x_i-x_{i'}\|^2}{\sigma^2_{\text{el-el}}}}}\Omega^{(l,\text{el, nuc})}_{i'k'pcs'}\Bigg) \\ &\times\Bigg(\sum_{i'} \frac{e^{-\frac{\|x'_i-R_k\|^2}{\sigma^2_{\text{el-el}}}}}{1+\sum_{i'} e^{-\frac{\|x_{i'}-R_k\|^2}{\sigma^2_{\text{el-nuc}}}}}\Omega^{(l,\text{el, nuc})}_{ik'pcs''}\Bigg)\Big]\end{aligned} \tag{5.26}$$

5.4.3 Fusion block

To construct the final interaction representation, we aggregate outputs from the three towers described above: $\Omega^{(L_{\text{el,nuc}})}_{ijpcs}$ denote the output of the local geometric product tower, $\Omega^{(L_{\text{orb,nuc}})}_{jkpcs}$ denote the output of orbital-nuclei MHSA, and $\Omega^{(L_{\text{el, orb}})}_{ijpcs}$ denote the output of electron-orbital MHSA tower. Each tensor is indexed over electrons (i), orbitals (j), nuclei (k), parity (p), channel (c), and hidden dimension (s). Our attention towers are permutation equivariant along the sequence axis (electrons or nuclei), while treating the orbital index j as a batch dimension. This ensures that orbital representations remain independent across electrons and nuclei as electron and nuclear dependencies are essential for prediction. This reduces computational overhead and maintains quadratic memory complexity in system size. However, to model full three-body correlations involving electrons, orbitals, and nuclei to produce final orbitals indexed by Φ_{ijs}, we construct a fused representation tensor Γ_{ijks} by contracting across all three embeddings. This operation requires calculating a joint tensor over (i, j, k) triplets, resulting in a cubic memory complexity with respect to the system size. This part is standard among all architechtures like FermiNet [Pfau et al., 2020b].

To make calculations more tractable, we first eliminate the Clifford structure by projecting over the (p, c) indices. This approach draws inspiration from the tensor train decomposition [Oseledets, 2011] principles.

$$Q^{(\text{el,nuc})}_{iks} = \text{LayerNorm}(\sum_{s'} W^{(1)}_{spcs'} \Omega^{(\text{el, orb})}_{ikpcs'} + b^{(1)}_s) \tag{5.27}$$

$$Q^{(\text{orb,nuc})}_{jks} = \text{LayerNorm}(\sum_{s'} W^{(2)}_{spcs'} \Omega^{(\text{orb, nuc})}_{jkpcs'} + b^{(2)}_s) \tag{5.28}$$

$$Q^{(\text{el,orb})}_{ijs} = \text{LayerNorm}(\sum_{s'} W^{(3)}_{spcs'} \Omega^{(\text{el, orb})}_{ijpcs'} + b^{(3)}_s) \tag{5.29}$$

We then construct a fused representation tensor Γ_{ijks} using a combination of the above projections:

$$\Gamma_{ijks} = \text{SiLU}(\sum_{s'} W^{(4)}_{ss'} \Omega^{(\text{el, orb})}_{ikpcs'} + \sum_{s'} W^{(5)}_{ss'} \Omega^{(\text{el, orb})}_{ijpcs'} + \sum_{s'} W^{(6)}_{ss'} \Omega^{(\text{orb, nuc})}_{jks'} + b^{(4)}_s) \tag{5.30}$$

5.4.4 Initial anti(symmetric) and time embeddings

To construct a physically valid fermionic wave function, we explicitly separate the learned representation into antisymmetric and symmetric components. Specifically, the antisymmetric embedding a_θ is antisymmetric with respect to permutations of electrons (including spin), and the symmetric embedding s_θ is symmetric. This separation reflects the dual nature of quantum states: the antisymmetric part enforces the Pauli exclusion principle, while the symmetric part captures exchange-invariant structures such as correlation, pairwise distances or external fields.

From the fused representation, we compute feature maps used in orbital envelope construction:

$$\sigma_{ijks} = Z_k \exp(\sum_{s'} W^{(7)}_{ss'} \Gamma_{ijks'} + b^{(7)}_s) \tag{5.31}$$

$$\rho_{ijks} = \sum_{s'} W^{(8)}_{ss'} \Gamma_{ijks'} + b^{(8)}_s \tag{5.32}$$

$$\vartheta_{ijks} = \sum_{s'} W^{(9)}_{ss'} \Gamma_{ijks'} + b^{(9)}_s \tag{5.33}$$

Using these quantities, we compute the complex-valued orbitals:

$$\Phi_{ijs} = \sum_{k} \exp(-\sigma_{ijks}\big(\sqrt{\|x_i - R_k\|^2_{\mathbb{R}^3} + 1} - 1\big) + \rho_{ijks} + i\vartheta_{ijks}) \tag{5.34}$$

The initial antisymmetric embedding is then defined as the determinant over these orbitals:

$$a^{(0)}_s = \det[\Phi_{ijs}]. \tag{5.35}$$

Note that this determinant operation is used once to initialize the antisymmetric embedding.

To obtain symmetric embedding, we perform reductions over the orbital, electron, and nucleus axes using permutation-invariant DeepSet-style architecture [Zaheer et al., 2017]:

$$s_s^{(0)} = \mathrm{LayerNorm}(\sum_i \mathrm{SiLU}(W_{ss'''}^{(12)} \mathrm{LayerNorm}(\sum_j \mathrm{SiLU}(\sum_{s'} W_{s'''s''}^{(11)} \\ \mathrm{LayerNorm}(\sum_k \mathrm{SiLU}(\sum_{s'} W_{s''s'}^{(10)} F_{ijks'} + b_{s''}^{(10)})) + b_{s'''}^{(11)})) + b_s^{(12)})) \tag{5.36}$$

This procedure enforces invariance to permutations across electrons i, nuclei j, and orbitals k, ensuring that $s_s^{(0)}$ captures exchange-invariant features of the full system.

Lastly, time-dependence is incorporated through a simple transformation:

$$\tau^{(0)}(t) = \mathrm{ReLU}(Wt + b), \tag{5.37}$$

where W and b are learnable parameters.

5.4.5 Sign-equivariant mixing layer and wave function construction

The mixing layer $f_\theta^{\mathrm{mix}} : \mathbb{R}^{d_{\mathrm{sym}}} \times \mathbb{C}^{asym} \to \mathbb{R}^{d_{\mathrm{sym}}} \times \mathbb{C}^{asym}$ satisfies a sign-equivariance constraint: $f_\theta^{\mathrm{mix}}(x, -y) = (f_\theta^{\mathrm{mix}}(x, y)_{\mathrm{sym}}, -f_\theta^{\mathrm{mix}}(x, y)_{\mathrm{asym}})$. This enforces that the antisymmetric part flips sign when the input flips sign, while the symmetric part stays invariant. Rather than requiring determinants to serve as universal approximators of arbitrary antisymmetric functions, which may be expensive or inefficient, we use the following idea. It suffices to ensure that the antisymmetric component distinguishes between physically distinct configurations. That is, for any two electron configurations $x_1, x_2 \in \mathbb{R}^3$, which are neither related by permutation nor contain identical coordinates (i.e., electron collisions), we require that the antisymmetric mapping satisfies $f_{\mathrm{asym}}(x_1) \neq f_{\mathrm{asym}}(x_2)$. This relaxed separability condition is sufficient to guarantee universal approximation given that the sign-equivariant transformation f_θ^{mix} is sufficiently expressive.

After obtaining the initial tuple of symmetric, antisymmetric and time embeddings,

$(s^{(0)}, a^{(0)}, \tau^{(0)})$, we define the initial ansatz as

$$\Psi_\theta^{(0)} = \sum_i e^{\sum_j w_{ij}^{(0)} s_j^{(0)}} a_i^{(0)}, \tag{5.38}$$

where w_{ij} are learnable parameters. Then, for every layer $l = 1, \ldots, L_{\text{mix}}$ we repeat:

$$\begin{aligned} h^{(l)} = \text{LayerNorm}([\text{SiLU}(\sum_j W_{ij}^{(l,1)} s_j^{(l-1)} + b_i^{(l,1)}) - \text{SiLU}(\sum_j W_{ij}^{(l,2)} (a_j^{(l-1)})^2 + b_i^{(l,2)})) \\ + \sin(\sum_j W_{ij}^{(l,3)} \tau_j^{(l-1)} + b_i^{(l,3)})]_{i=1}^{d_{\text{hid}}}) \end{aligned} \tag{5.39}$$

Note that the squaring $a^{(l-1)}$ effectively eliminates sign information, thereby transforming the antisymmetric function into a symmetric one. This transformation makes it compatible with the symmetric embedding, ensuring that $h^{(l)}$ inherently maintains symmetric properties. The construction of the intermediate matrix H^l goes as follows:

$$\begin{aligned} H^l = [[\text{SoftPlus}(\sum_k (W_{ijk}^{(l,4)} + W_{jik}^{(l,4)}) h_k^{(l)}) (\cos(\sum_k (W_{ijk}^{(l,5)} - W_{jik}^{(l,5)}) h_k^{(l)}) \\ + i \sin(\sum_k (W_{ijk}^{(l,5)} + W_{jik}^{(l,5)}) h_k^{(l)}))]_{j=1}^{d_{\text{asym}}}]_{i=1}^{d_{\text{asym}}} \end{aligned} \tag{5.40}$$

By construction, $H_{ij}^{(l)}$ satisfies the skew-Hermitian property:

$$H_{ij}^{(l)} = -\overline{H_{ji}^{(l)}}.$$

The updated antisymmetric embedding is computed as:

$$a^{(l)} = [a_i^{(l-1)} + \sum_j H_{ij}^{(l)} a_j^{(l-1)}]_{i=1}^{d_{\text{asym}}}. \tag{5.41}$$

Note that while $H_{ij}^{(l)}$ is guaranteed to be non-zero, it can approach arbitrarily close to zero. This choice of an anti-Hermitian structure ensures orthogonality between successive antisymmetric embeddings $a^{(l)} \perp a^{(l-1)}$, enforcing each layer of the mixer to learn novel features without redundancy. The updated symmetric embedding is computed as:

$$s^{(l)} = [s^{(l-1)} + \mathrm{SiLU}(W_{ij}^{(l,6)} h_j^{(l)} + b_i^{(l,6)})]_{i=1}^{d_{\mathrm{sym}}}, \tag{5.42}$$

the new time embedding:

$$\tau^{(l)} = [\tau^{(l-1)} + \mathrm{ReLU}(W_{ij}^{(l,7)} \tau_j^{(l-1)} + b_i^{(l,7)}]_{i=1}^{d_{\mathrm{time}}}). \tag{5.43}$$

Finally, we refine the ansatz:

$$\Psi_\theta^{(l)} = \Psi_\theta^{(l-1)} + \sum_i e^{\sum_j w_{ij}^{(l)} s_j^{(l)}} a_i^{(l)}. \tag{5.44}$$

After repeating this procedure L_{mix} times, we define $\Psi_\theta := e^{J(x)} \Psi_\theta^{(L_{\mathrm{mix}})} \in \mathbb{C}$, where $e^{J(x,R,Z,S)}$ is a Jastrow factor.

Cusp conditions To ensure proper physical behavior near potential singularities, our approach explicitly incorporates both electron-electron and electron-nuclei cusp conditions into the wave function ansatz. Following von Glehn et al. [2022], our Jastrow factor takes the following form:

$$J_\theta(x, R, Z, S) = \sum_{i<j; S_i = S_j} -\frac{1}{4} \frac{\alpha_{\mathrm{par}}^2}{\alpha_{\mathrm{par}} + \|x_i - x_j\|} + \sum_{i<j; S_i \neq S_j} -\frac{1}{2} \frac{\alpha_{\mathrm{anti}}^2}{\alpha_{\mathrm{anti}} + \|x_i - x_j\|}, \tag{5.45}$$

which has two free parameters α_{par} and α_{anti} for same-spin and opposite-spin electron pairs, respectively.

5.5 Experiments

To validate our stochastic fermionic framework, we simulate the quantum dynamics of a hydrogen molecule H_2. The system is governed by the molecular Hamiltonian:

$$\begin{aligned} H &= H^{\text{kin}} + H^{\text{el-el}} + H^{\text{el-nuc}} + H^{\text{nuc-nuc}} \\ &= -\frac{1}{2}\sum_{i=1}^{n_{\text{el}}} \nabla_i^2 + \sum_{i=1}^{n_{\text{el}}} \sum_{j=i+1}^{n_{\text{el}}} \frac{1}{\|x_i - x_j\|} - \sum_{i=1}^{n_{\text{el}}} \sum_{I=1}^{n_{\text{nuc}}} \frac{Z_I}{\|x_i - R_I\|} + \sum_{I=1}^{n_{\text{nuc}}} \sum_{J=I+1}^{n_{\text{nuc}}} \frac{Z_I Z_J}{\|R_I - R_J\|}, \end{aligned} \tag{5.46}$$

where $n_{\text{el}} = 2$ is the number of electrons, $n_{\text{nuc}} = 2$ denotes the number of nuclei, Z_I represents the atomic number ($\sum_{j=1}^{n_{\text{nuc}}} Z_I = n_{\text{el}}$), x_i and R_I are the positions of electrons and nuclei, respectively. All coordinates are given in $\mathbb{R}^3$.

The initial state $\Psi_0 = \Psi(x, t = 0)$ is given as a superposition of the ground state ψ_0 and the first excited state ψ_1:

$$\Psi(x, t = 0) = c_0\psi_0(x) + c_1\psi_1(x), \tag{5.47}$$

where c_0 and c_1 are constants. The eigenstates ψ_0, ψ_1 with the corresponding energies E_0 and E_1 can be obtained through HF, FullCI, or any other method to evaluate these quantities. To initialize the simulation, we require samples from the initial probability density $\rho_0 = |\Psi(x, t = 0)|^2$. For this purpose, MCMC sampling can be used, for example, the Metropolis-Hastings algorithm [Metropolis et al., 1953].

5.5.1 Baseline

To establish a performance baseline for our approach, we consider the analytical solution of TDSE that can be obtained from solutions of the time-independent one. In particular, the TDSE solution can be constructed from the solutions of the time-independent Schrödinger equation by combining the eigenfunctions with a time-dependent phase factor. The exact time-evolved wave function is given by:

$$\Psi(x,t) = \sum_{n=0}^{1} c_n \psi_n(x) e^{-iE_n t/\hbar} \tag{5.48}$$

where E_n denotes the energy eigenvalue corresponding to eigenstate ψ_n. We use the FullCI method to estimate the ground state ψ_0 and the corresponding energy E_0, and the excited state ψ_1 and the corresponding energy E_1.

For practical computations, we do not store the full wave function, which would be prohibitively expensive due to the curse of dimensionality. Even for our H_2 system with two electrons in a three-dimensional space, the wave function $\Psi(x,t)$ is defined over $\mathbb{R}^6$. Discretizing this domain with 100 grid points per dimension would require storing $100^6 = 10^{12}$ values. Instead, we employ MCMC sampling to sample from the probability density $|\Psi(x,t)|^2$. This sampling-based approach enables efficient estimation of density without explicitly storing the full wave function.

The accuracy of this baseline implementation depends on two factors: (1) the fidelity of the initial condition sampling, and (2) the efficiency of the MCMC sampler used to generate samples from the corresponding density. These factors establish methodological constraints that must be considered when evaluating the performance of our proposed framework.

5.5.2 Experimental results

We present results from three distinct approaches: (1) the baseline – the eigenstate expansion method with MCMC sampling, (2) F-DSM with initial conditions obtained with FullCI, and (3) F-DSM with initial conditions obtained with our pretrained model $\Psi_\theta^{\mathrm{IC}}$.

Density For visualization purposes, we project the full electron density onto a one-dimensional subspace to capture the essential quantum dynamics. While the complete electronic configuration space for our H_2 molecular system exists in $\mathbb{R}^6$ (representing two electrons in 3d space), the system's primary dynamics occurs along the first spatial dimension. Consequently, we visualize only this dimension, which captures the essential behavior of our system.

Figure 5.2 presents an evolution of the probability density distributions obtained from the three methods. The baseline has a pronounced periodic behavior. This oscillatory pattern represents the quantum mechanical dynamics expected in such systems. The F-DSM framework with FullCI-derived initial conditions produces a different result with a relatively static density distribution that lacks the oscillatory features. This significant deviation suggests that, while FullCI is supposed to provide excellent energy estimates at $t = 0$, its poor representation of important features for dynamics leads to substantial errors when propagating in time. In contrast, the F-DSM framework with our pretrained model $\Psi_\theta^{\mathrm{IC}}$ demonstrates promising consistency with the baseline's oscillatory behavior and preserves the essential time-dependent dynamics of the electron density. This improvement over the FullCI-based F-DSM approach highlights the importance of a proper wave function representation for accurate quantum dynamics simulations.

Furthermore, we evaluate the accuracy of each method by computing the TDSE error, defined as $\|i\hbar\partial_t\Psi(x,t) - \mathcal{H}\Psi(x,t)\|_2$. This measures how well the predicted wavefunction satisfies the fundamental equation governing quantum dynamics. The baseline eigenstate expansion method demonstrates consistency with the TDSE error of 1.12. The F-DSM approach with the IC coming from FullCI shows substantially higher deviation with an error of 15.16. Our F-DSM with the pretrained model $\Psi_\theta^{\mathrm{IC}}$ achieves better consistency with a TDSE error of only 0.47, outperforming even the baseline method. These results demonstrate that our F-DSM framework captures the essential structure of the quantum system, leading to better numerical precision than conventional methods.

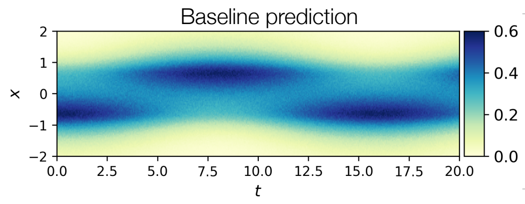
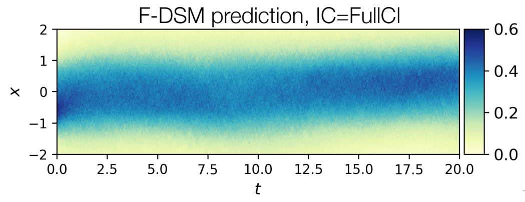
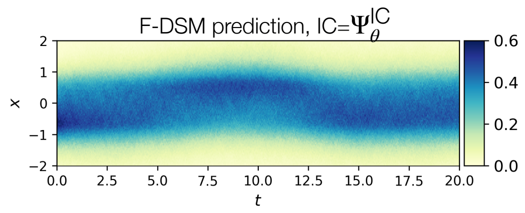
	TDSE error	Sampling time (s/time step)
	1.12	6.24
	15.16	**0.77**
	**0.47**	1.27

Figure 5.2: Comparison of quantum dynamics simulation methods for the H_2 molecule. The left column shows time evolution of electronic density for three approaches: the baseline, F-DSM with FullCI initial conditions, and F-DSM with pretrained model $\Psi_\theta^{\mathrm{IC}}$ initial conditions. The right columns quantify performance metrics: TDSE error and sampling time per time step. The F-DSM approach with a pretrained model achieves the lowest TDSE error while maintaining a competitive sampling speed (1.27 s/iter). Bold values indicate the best performance for each metric.

Computational time We also evaluate the computational efficiency of our approach against the baseline, as shown in Figure 5.2. Using identical parameters (detailed in Appendix D.1) and 65,536 samples per time step across all methods, we observe significant performance differences. The baseline eigenstate expansion method requires 6.24 seconds per time step, while our F-DSM variants demonstrate substantial speedups. Our F-DSM with the pretrained model approach takes 1.27 seconds per time step, nearly 5× faster than the baseline. The F-DSM with FullCI initial conditions achieves even better performance at 0.77 seconds per time step. It is not surprising since it uses a smaller basis set representation instead of evaluating the neural network. These speedups arise from our stochastic reformulation that avoids repeated MCMC sampling at each time step. This computational advantage is expected to become even more pronounced for larger molecular systems, where MCMC sampling costs increase rapidly with dimensionality.

Cusp behavior and initial condition quality The performance of our pretrained model can be directly attributed to its accurate reproduction of electron cusps. Figure 5.3 compares the initial condition distributions obtained from FullCI and our pretrained model $\Psi_\theta^{\mathrm{IC}}$. While FullCI is expected to achieve high energy accuracy, its Gaussian basis set inherently smooths over the non-differentiable cusps required by Kato's condition.

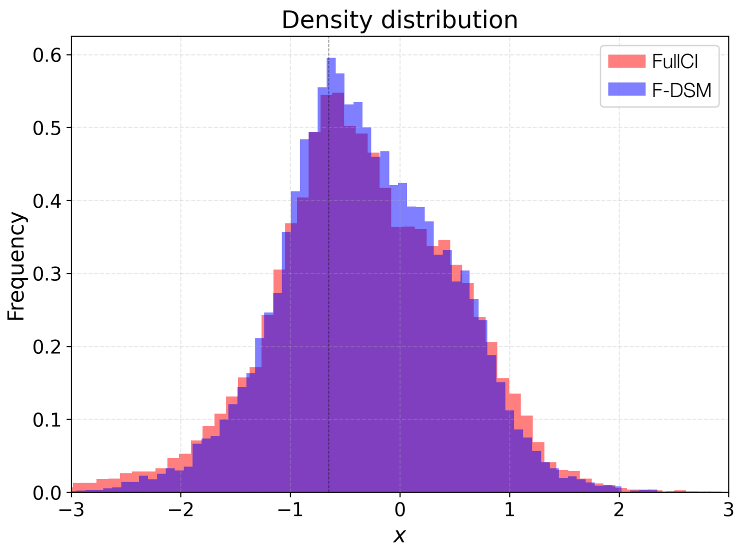


Figure 5.3: Initial condition distribution predicted by the pretrained model $\Psi_\theta^{(\mathrm{IC})}$ and FullCI. Our pretrained model $\Psi_\theta^{(\mathrm{IC})}$ captures sharp features near electron–nucleus coalescence point, consistent with Kato's cusp conditions, while FullCI smooths over these regions due to basis set limitations.

These cusps are not merely mathematical formalities – they precisely counterbalance Coulomb singularities in the potential, preventing the kinetic energy from diverging at coalescence points. Our neural architecture explicitly enforces these constraints, ensuring the wave function's gradient discontinuity exactly cancels the potential's singularity. This likely explains why our pretrained model achieves lower TDSE error despite starting from the same mathematical formalism as the FullCI-initialized version.

Energy observables Energy plays a fundamental role in quantum chemistry, serving not only as a physical observable but also as a principal benchmark for evaluating the quality of approximate wave functions. The variational principle ensures that methods seeking approximate solutions to the Schrödinger equation inherently aim to minimize the energy, making energy evaluations a natural metric for methodological comparisons. In this context, it is important to evaluate not only the mean energy but also its variance, as the latter directly reflects numerical stability and precision of simulations. High variance in energy estimates means inaccuracies or numerical instabilities, leading to less reliable predictions and limiting the practical utility of computational approaches.

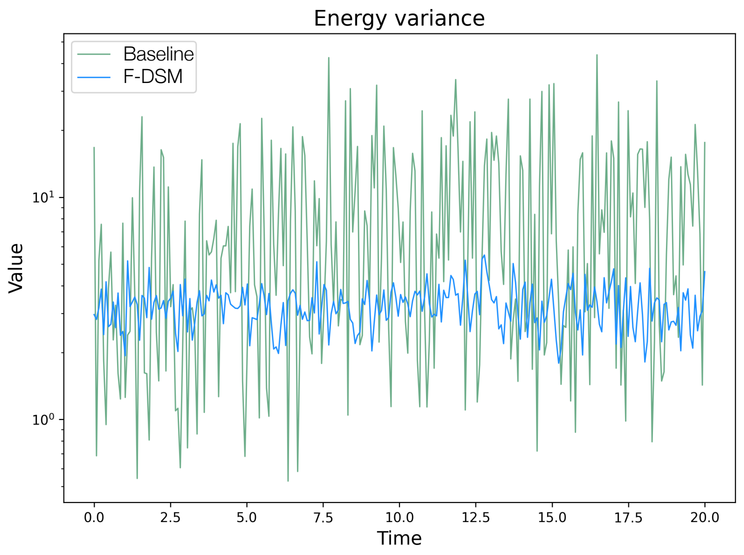


Figure 5.4: Comparison of energy variance for the baseline and F-DSM models. The F-DSM approach exhibits significantly lower variance compared to the FullCI-based baseline. This highlights the improved numerical stability and robustness of our approach, even without explicit energy minimization during training.

We evaluate the performance of our approach by comparing energy variance against the baseline method. While FullCI directly optimizes energy in its formulation, our F-DSM framework does not explicitly incorporate energy minimization during model training. Figure 5.4 illustrates the energy variance observed for both methods throughout the simulation period. Notably, F-DSM exhibits substantially lower variance compared to the baseline, differing by orders of magnitude. This difference in performance can be attributed to our ansatz's capacity to efficiently model the wave function structure, particularly at nuclear cusps and nodal surfaces where traditional methods struggle. This performance demonstrates that our F-DSM formulation can serve as an effective framework for capturing fundamental quantum dynamics even without explicit energy constraints. These findings suggest that F-DSM offers a compelling alternative to conventional electronic structure methods for time-dependent quantum simulations.

Method	Mean Energy (Ha)	Relative Abs Error (%)
Reference	-1.1347	–
Baseline	-1.1537	1.67
F-DSM	-1.1389	**0.37**

Table 5.1: Mean energy values and relative absolute errors for the hydrogen molecule. The reference value is compared with the baseline model and our proposed F-DSM approach, which achieves significantly lower error. Bold values indicate the best performance.

As for the energy itself, Table 5.1 highlights these results. The reference value is known: -1.1347 Ha. The temporal mean energies obtained by the baseline and F-DSM are -1.1537 Ha and -1.1389 Ha, yielding relative absolute errors ($|\frac{E-\hat{E}}{E}| \times 100\%$) of 1.674% and 0.370% respectively. This high accuracy, combined with the previously discussed variance reduction, highlights the efficacy of our ansatz parametrization in capturing essential electronic wave function features.

5.6 Conclusion

We introduce a novel framework for simulating time-dependent fermionic quantum dynamics based on the DSM framework. Our work makes several contributions: we develop an antisymmetric ansatz that effectively combines antisymmetric and symmetric representations through a carefully designed sign-equivariant mixer layer, addressing the fundamental challenge of maintaining fermionic antisymmetry in neural representations. By incorporating Clifford-algebra-based encodings, our method naturally embeds essential geometric and algebraic structures that are critical for quantum systems. Furthermore, we systematically enforce key physical constraints such as cusp conditions, ensuring physically meaningful wave functions throughout the simulation. Empirical results on H_2 molecular dynamics demonstrate promising performance, with our model showing improved numerical stability and reasonable accuracy in capturing electron dynamics, even in the presence of excited states. These findings highlight the promise of F-DSM as an efficient alternative for time-dynamics simulations in quantum chemistry.

5.7 Discussion and limitations

While our framework shows strong performance in capturing essential quantum dynamics, several limitations remain. Our current results focus on the H_2 molecule, which, while challenging due to its excited state dynamics, represents one of the simplest molecular systems. Testing on larger molecules with more complex electron correlation patterns is necessary to establish the generalizability of our method. While the F-DSM framework offers promising computational advantages, we have not yet extensively tested how the method scales with increasing system size and complexity. Besides, the performance of our approach relies on the quality of initial conditions. A detailed analysis of initialization challenges using different quantum chemistry methods is presented in Appendix D.2. A comprehensive evaluation of how pretraining and possibly other methods to obtain the ICs affect the performance of our method across different molecular systems would require a significant additional research effort. Another interesting direction – benchmarking the pretrained model component of our framework against state-of-the-art ground state methods. While our current work focuses on time-dependent dynamics, the novel neural architecture for representing antisymmetric wave functions could potentially advance ground state calculations as well. This area still remains central to computational quantum chemistry. By quantifying the accuracy and computational efficiency of this approach for ground state problems, we could establish F-DSM as a versatile framework spanning both ground state and time evolution simulations.

CHAPTER 6
CONCLUSION

This dissertation explores the application of machine learning techniques across three scientific domains: subseasonal weather forecasting, emulation of chaotic dynamics, and quantum mechanics simulations for bosons and fermions. Each contribution (Chapter 2-5) demonstrates how carefully designed machine learning frameworks can address long-standing methodological and computational bottlenecks while preserving essential physical properties. Though specific future directions are noted throughout the thesis, we outline some of them here.

Chapter 2 presents a systematic exploration of *ML methods for subseasonal forecasting*, a domain characterized by the challenging prediction window between short-term and long-term climate forecasts. Our investigation demonstrates that significant predictive skill improvements can be achieved by extracting information embedded in physics-based ensemble forecasts. We develop a new framework incorporating three complementary innovations: (1) direct utilization of the complete ensemble distribution to capture the probabilistic nature and internal variability of these predictions, (2) employing positional encoding and convolutional neural networks to model complex spatial dependencies, (3) development of the model stacking framework that combines multiple spatial modeling paradigms. Furthermore, we introduce quantile regression, an approach underutilized in climate sciences, to predict extreme events. These advances provide valuable insights for improving the predictive skill for both temperature and precipitation at the challenging subseasonal timescale. *Future research directions* in this domain should explore utilizing ensemble forecasts from multiple prediction centers simultaneously as different organizations employ different initialization strategies and parameterizations. Two methodological challenges require particular attention: advanced uncertainty quantification beyond our quantile regression approach, potentially through conformalized prediction techniques, and development of methods robust to a distribution shift between hindcast and forecast data in changing climate.

Chapter 3 introduces *novel methodologies for training neural operators on chaotic dynamical systems* through two complementary approaches: optimal transport and contrastive learning. An optimal transport method that leverages system knowledge to match summary statistics, and a contrastive learning strategy that directly learns invariant statistics through multi-environment training. By focusing on preserving time-invariant statistical properties of of chaotic attractors rather than short-term prediction accuracy, our methods enable neural operators to capture long-term statistical behavior even in high-noise regimes in contrast to traditional RMSE-focused training. On a variety of chaotic systems, our method is shown empirically to preserve invariant measures of chaotic attractors. The immediate *future directions* include extending these methods to handle time-dependent systems by developing techniques that compute statistics over restricted temporal windows, enabling the study of slowly varying dynamics and discrete transitions such as tipping points. We plan to investigate applications to SDEs and stochastic PDEs. A deeper investigation of the trade-offs between short-term forecasting accuracy and long-term statistical fidelity (invariant statistics) across different neural architectures is important for advancing the field.

Chapter 4 introduces *Deep Stochastic Mechanics (DSM), a novel approach to quantum simulation* that addresses the curse of dimensionality by leveraging the latent low-dimensional structure of quantum systems. Instead of directly solving the Schrödinger equation, DSM establishes a correspondence between the Schrödinger equation and a diffusion process through a stochastic interpretation of quantum mechanics. By learning the drift terms of these stochastic differential equations using neural networks, we enable efficient sampling from quantum probability densities without explicitly computing wave functions. This sampling strategy avoids MCMC as this stochastic formulation allows to sample particle trajectories directly based on the SDE. Our approach demonstrates favorable quadratic computational scaling with the problem dimension, in contrast to traditional grid-based methods with their exponential scaling. This DSM approach is evaluated on different noninteracting and interacting bosonic systems, showing high accuracy in predicting quantum dynamics and performing better than other conventinal methods. *Future research* should focus on extending DSM beyond the current implementation for bosons (our work in progress). Other directions include developing higher-order SDE integrators to improve computational efficiency, and applying the framework to more complex quantum systems including those with relativistic effects through Dirac and Klein-Gordon equations.

Chapter 5 *extends original DSM framework to fermionic systems, developing Fermionic Deep Stochastic Mechanics (F-DSM)* to address the unique challenges posed by antisymmetric wave functions and the Pauli exclusion principle. Our framework has three key innovations: (1) a paradigm-shifting neural network architecture that represents antisymmetric wave functions through nonlinear compositions rather than traditional linear combinations of antisymmetric functions, (2) a novel orbital positional encoding based on Clifford algebra that efficiently parameterizes electronic structure with system-size independent parameters, and (3) while the fundamental stochastic correspondence remains, our approach addresses critical numerical challenges in fermionic simulations through a specialized two-phase training procedure that ensures proper representation of singular features in electronic wave functions. Experiments with a hydrogen molecule demonstrate that F-DSM can reasonably simulate its dynamics even when starting from the excited state. *Future developments* for F-DSM will focus on extending the framework to larger molecular systems with more complex electronic structures, and incorporating additional quantum mechanical effects such as spin-orbit coupling and relativistic corrections. Another interesting direction is benchmarking the pretrained model component of our framework against state-of-the-art ground state methods. While our current work focuses on time-dependent dynamics, our novel neural architecture for representing antisymmetric wave functions could potentially advance ground state calculations as well. By quantifying the accuracy and computational efficiency of this approach for ground state problems, we could establish F-DSM as a versatile framework spanning both ground state and time evolution simulations.

REFERENCES

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

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

Alex Andonian, Taesung Park, Bryan Russell, Phillip Isola, Jun-Yan Zhu, and Richard Zhang. Contrastive feature loss for image prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1934–1943, 2021.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/arjovsky17a.html.

Brian M Austin, Dmitry Yu Zubarev, and William A Lester Jr. Quantum Monte Carlo and related approaches. *Chemical reviews*, 112(1):263–288, 2012.

Shayan Aziznejad, Harshit Gupta, Joaquim Campos, and Michael Unser. Deep neural networks with trainable activations and controlled Lipschitz constant. *IEEE Transactions on Signal Processing*, 68:4688–4699, 2020.

Hamada S Badr, Benjamin F Zaitchik, and Seth D Guikema. Application of statistical models to the prediction of seasonal rainfall anomalies over the sahel. *Journal of Applied meteorology and climatology*, 53(3):614–636, 2014.

M Bajdich, L Mitas, G Drobnỳ, LK Wagner, and KE Schmidt. Pfaffian pairing wave functions in electronic-structure quantum Monte Carlo simulations. *Physical review letters*, 96 (13):130201, 2006.

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

R. J Ban, C. M. Bitz, A. Brown, E. Chassignet, J. A. Dutton, R. Hallberg, A. Kamrath, D. Kleist, P. F. J. Lermusiaux, H. Lin, L. Myers, J. Pullen, S. Sandgathe, M. Shafer, D. Waliser, and C. Zhang. *Next Generation Earth System Prediction: Strategies for Subseasonal to Seasonal Forecasts.* The National Academies Press, 2016.

Stefano Barison, Filippo Vicentini, and Giuseppe Carleo. An efficient quantum algorithm for the time evolution of parameterized circuits. *Quantum*, 5:512, 2021.

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

Anthony G Barnston, Michael K Tippett, Michelle L L'Heureux, Shuhua Li, and David G DeWitt. Skill of real-time seasonal enso model predictions during 2002–11: Is our capability increasing? *Bulletin of the American Meteorological Society*, 93(5):631–651, 2012.

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

Daniel J Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549(7671):188–194, 2017.

Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997. doi:10.1137/S0097539796300921.

A Beskos, N Pillai, G Roberts, J. Sanz-Serna, and A Stuart. Optimal tuning of the hybrid Monte Carlo algorithm. *Bernoulli*, 19, 11 2013.

Michael Beyer and Wolfgang Paul. Stern–Gerlach, EPRB and Bell inequalities: An analysis using the quantum Hamilton equations of stochastic mechanics. *Foundations of Physics*, 54(2):20, 2024.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Pangu-Weather: A 3d high-resolution model for fast and accurate global weather forecast. *arXiv preprint arXiv:2211.02556*, 2022.

Gérard Biau and Erwan Scornet. A random forest guided tour. *Test*, 25:197–227, 2016.

Ph Blanchard, Ph Combe, M Sirugue, and M Sirugue-Collin. 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*, pages 65–86. Springer, 2005.

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

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

David Bohm. 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`.

George H Booth and Ali Alavi. Approaching chemical accuracy using full configuration-interaction quantum Monte Carlo: A study of ionization potentials. *The Journal of chemical physics*, 132(17), 2010.

Max Born and W Heisenberg. Zur quantentheorie der molekeln. *Original Scientific Papers Wissenschaftliche Originalarbeiten*, pages 216–246, 1985.

Jonah Botvinick-Greenhouse, Robert Martin, and Yunan Yang. Learning dynamics on invariant measures using PDE-constrained optimization. *Chaos*, 33(6), 1 June 2023. URL http://dx.doi.org/10.1063/5.0149673.

Johannes Brandstetter, Rianne van den Berg, Max Welling, and Jayesh K Gupta. Clifford neural layers for PDE modeling. *arXiv preprint arXiv:2209.04934*, 2022a.

Johannes Brandstetter, Max Welling, and Daniel E Worrall. Lie point symmetry data augmentation for neural PDE solvers. In *International Conference on Machine Learning*, pages 2241–2256. PMLR, 2022b.

Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural PDE solvers. In *International Conference on Learning Representations*, 2022c. URL https://openreview.net/forum?id=vSix3HPYKSU.

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

Gilbert Brunet, Melvyn Shapiro, Brian Hoskins, Mitch Moncrieff, Randall Dole, George N Kiladis, Ben Kirtman, Andrew Lorenc, Brian Mills, Rebecca Morss, et al. Collaboration of the weather and climate communities to advance subseasonal-to-seasonal prediction. *Bulletin of the American Meteorological Society*, 91(10):1397–1406, 2010.

Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

Rainer Buckdahn, Juan Li, Shige Peng, and Catherine Rainer. 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.

Eric Cances, Mireille Defranceschi, Werner Kutzelnigg, Claude Le Bris, and Yvon Maday. Computational quantum chemistry: a primer. *Handbook of numerical analysis*, 10:3–270, 2003.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

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

Giuseppe Carleo, Lorenzo Cevolani, Laurent Sanchez-Palencia, and Markus Holzmann. 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.

Giuseppe Carleo, Matthias Troyer, Giacomo Torlai, Roger Melko, Juan Carrasquilla, and Guglielmo Mazzola. Neural-network quantum states. *Bulletin of the American Physical Society*, 63, 2018.

Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová. Machine learning and the physical sciences. *Reviews of Modern Physics*, 91(4):045002, 2019.

Benjamin A Cash, Julia V Manganello, and James L Kinter. Evaluation of NMME temperature and precipitation bias and forecast skill for South Asia. *Climate dynamics*, 53: 7363–7380, 2019.

MA Cazalilla and JB Marston. Time-dependent density-matrix renormalization group: A systematic method for the study of quantum many-body out-of-equilibrium systems. *Physical review letters*, 88(25):256403, 2002.

D Ceperley. Ground state of the fermion one-component plasma: A Monte Carlo study in two and three dimensions. *Physical Review B*, 18(7):3126, 1978.

Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12299–12310, 2021.

Liuyi Chen, Bocheng Han, Xuesong Wang, Jiazhen Zhao, Wenke Yang, and Zhengyi Yang. Machine learning methods in weather and climate applications: A survey. *Applied Sciences*, 13(21):12019, 2023.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

Yuming Chen, Daniel Sanz-Alonso, and Rebecca Willett. Autodifferentiable ensemble kalman filters. *SIAM Journal on Mathematics of Data Science*, 4(2):801–833, 2022.

Claude C Chevalley. *The algebraic theory of spinors*. Columbia University Press, 1954.

Jiří Čížek. On the correlation problem in atomic and molecular systems. Calculation of wavefunction components in Ursell-type expansion using quantum-field theoretical methods. *The Journal of Chemical Physics*, 45(11):4256–4266, 1966.

Jiří Čížek. On the use of the cluster expansion and the technique of diagrams in calculations of correlation effects in atoms and molecules. *Advances in chemical physics*, 14:35–89, 1969.

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

William Kingdon Clifford. Preliminary sketch of bi-quaternions. *London Mathematical Society*, 4, 1873.

William Kingdon Clifford. *Mathematical Papers by William Kingdon Clifford: Edited by Robert Tucker, with an introduction by HJ Stephen Smith*. Macmillan and Company, 1882.

Antonio S Cofıno, Rafael Cano, Carmen Sordo, and Jose M Gutierrez. Bayesian networks for probabilistic weather prediction. In *15th Eureopean Conference on Artificial Intelligence (ECAI)*, 2002.

Judah Cohen, Dim Coumou, Jessica Hwang, Lester Mackey, Paulo Orenstein, Sonja Totz, and Eli Tziperman. S2s reboot: An argument for greater inclusion of machine learning in subseasonal to seasonal forecasts. *Wiley Interdisciplinary Reviews: Climate Change*, 10 (2):e00567, 2019.

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

J. F. Corney and P. D. Drummond. 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`.

Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(9):10850–10869, 2023.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26, Red Hook, NY, 2013. Curran Associates, Inc. URL `https://proceedings.neurips.cc/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf`.

Thaddeus George Dankel. Mechanics on manifolds and the incorporation of spin into Nelson's stochastic mechanics. *Archive for Rational Mechanics and Analysis*, 37:192–221, 1970.

Jean-Pierre Daudey, Jean-Louis Heully, and Jean-Paul Malrieu. Size-consistent self-consistent truncated or selected configuration interaction. *The Journal of chemical physics*, 99(2):1240–1254, 1993.

Peter Davidson. *Turbulence: An Introduction for Scientists and Engineers*. Oxford University Press, 06 2015. ISBN 9780198722588. doi:10.1093/acprof:oso/9780198722588.001.0001.

GF De Angelis and Maurizio Serva. Jump processes and diffusions in relativistic stochastic mechanics. In *Annales de l'IHP Physique théorique*, volume 53, pages 301–317, 1990.

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

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

Timothy DelSole and Michael K Tippett. Comparing forecast skill. *Monthly Weather Review*, 142(12):4658–4678, 2014.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009.

A Eugene DePrince III, Matthew Pelton, Jeffrey R Guest, and Stephen K Gray. Emergence of excited-state plasmon modes in linear hydrogen chains from time-dependent quantum mechanical methods. *Physical review letters*, 107(19):196806, 2011.

Maaneli Derakhshani and Guido Bacciagaluppi. On multi-time correlations in stochastic mechanics, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

Paul AM Dirac. Note on exchange phenomena in the Thomas atom. In *Mathematical proceedings of the Cambridge philosophical society*, volume 26, pages 376–385. Cambridge University Press, 1930.

J. R. Dorfman. *An Introduction to Chaos in Nonequilibrium Statistical Mechanics.* Cambridge Lecture Notes in Physics. Cambridge University Press, 1999. doi:10.1017/CBO9780511628870.

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

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

Thom H Dunning Jr. Gaussian basis sets for use in correlated molecular calculations. i. the atoms boron through neon and hydrogen. *The Journal of chemical physics*, 90(2): 1007–1023, 1989.

Pavel Dvurechensky, Alexander Gasnikov, and Alexey Kroshnin. Computational optimal transport: Complexity by accelerated gradient descent is better than by sinkhorn's algorithm. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1367–1376. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/dvurechensky18a.html`.

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

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

Yun Fan and Huug Van den Dool. A global monthly land surface air temperature analysis for 1948–present. *Journal of Geophysical Research: Atmospheres*, 113(D1), 2008.

Shi Zan Fang and Paul Malliavin. Stochastic analysis on the path space of a Riemannian manifold: I. Markovian stochastic calculus. *Journal of functional analysis*, 118(1):249–274, 1993.

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

Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré. Interpolating between optimal transport and mmd using sinkhorn divergences. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 2681–2690, Naha, Okinawa, Japan, 16–18 Apr 2019. PMLR. URL https://proceedings.mlr.press/v89/feydy19a.html.

Richard P Feynman and Michael Cohen. Energy spectrum of the excitations in liquid helium. *Physical Review*, 102(5):1189, 1956.

William MC Foulkes, Lubos Mitas, RJ Needs, and Guna Rajagopal. Quantum Monte Carlo simulations of solids. *Reviews of Modern Physics*, 73(1):33, 2001.

Jacov Frenkel. Waves mechanics: Advanced general theory. *The Mathematical Gazette (Oxford)*, 18(229):208–209, 1934. doi:10.2307/3606836.

Jaroslav Frnda, Marek Durica, Jan Rozhon, Maria Vojtekova, Jan Nedoma, and Radek Martinek. ECMWF short-term prediction accuracy improvement by deep learning. *Scientific Reports*, 12(1):7898, 2022.

John Cristian Borges Gamboa. Deep learning for time-series analysis, 2017.

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

Nicholas Gao and Stephan Günnemann. Neural Pfaffians: Solving many many-electron Schrödinger equations. *arXiv preprint arXiv:2405.14762*, 2024.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

R Gaudoin, M Nekovee, WMC Foulkes, RJ Needs, and G Rajagopal. Inhomogeneous random-phase approximation and many-electron trial wave functions. *Physical Review B*, 63(11):115115, 2001.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International conference on machine learning*, pages 1243–1252. PMLR, 2017.

Andrew Gelman, Walter R Gilks, and Gareth O Roberts. Weak convergence and optimal scaling of random walk Metropolis algorithms. *The annals of applied probability*, 7(1): 110–120, 1997.

Amir Ghaderi, Borhan M Sanandaji, and Faezeh Ghaderi. Deep forecast: Deep learning-based spatio-temporal forecasting, 2017.

Zoubin Ghahramani. Unsupervised learning. In *Summer school on machine learning*, pages 72–112. Springer, 2003.

H. S. Greenside, A. Wolf, J. Swift, and T. Pignataro. Impracticality of a box-counting algorithm for calculating the dimensionality of strange attractors. *Phys. Rev. A*, 25:3453–3456, Jun 1982. doi:10.1103/PhysRevA.25.3453. URL https://link.aps.org/doi/10.1103/PhysRevA.25.3453.

Andreas Griewank and Andrea Walther. *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.

Peter Grönquist, Chengyuan Yao, Tal Ben-Nun, Nikoli Dryden, Peter Dueben, Shigang Li, and Torsten Hoefler. Deep learning for post-processing ensemble weather forecasts. *CoRR*, abs/2005.08748, 2020. URL https://arxiv.org/abs/2005.08748.

T. H. Gronwall. 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.

Aditya Grover, Ashish Kapoor, and Eric Horvitz. A deep hybrid model for weather forecasting. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 379–386, 2015.

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

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

Jayesh K Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized pde modeling. *arXiv preprint arXiv:2209.15616*, 2022.

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

Jutho Haegeman, J Ignacio Cirac, Tobias J Osborne, Iztok Pižorn, Henri Verschelde, and Frank Verstraete. Time-dependent variational principle for quantum lattices. *Physical review letters*, 107(7):070601, 2011.

Brian C Hall and Brian C Hall. *Lie groups, Lie algebras, and representations*. Springer, 2013.

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

Jiequn Han, Yingzhou Li, Lin Lin, Jianfeng Lu, Jiefu Zhang, and Linfeng Zhang. Universal approximation of symmetric and anti-symmetric functions. *arXiv preprint arXiv:1912.01765*, 2019a.

Jiequn Han, Linfeng Zhang, and E Weinan. Solving many-electron Schrodinger equation using deep neural networks. *Journal of Computational Physics*, 399:108929, 2019b.

Shijie Hao, Yuan Zhou, and Yanrong Guo. A brief survey on semantic segmentation with deep learning. *Neurocomputing*, 406:302–321, 2020.

Zhongkai Hao, Chengyang Ying, Zhengyi Wang, Hang Su, Yinpeng Dong, Songming Liu, Ze Cheng, Jun Zhu, and Jian Song. Gnot: A general neural operator transformer for operator learning. *arXiv preprint arXiv:2302.14376*, 2023.

Douglas Rayne Hartree. The wave mechanics of an atom with a non-Coulomb central field. Part II. Some results and discussion. *Mathematical Proceedings of the Cambridge Philosophical Society*, 24(1):111–132, 1928. doi:10.1017/S0305004100011920.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Sue Ellen Haupt, William Chapman, Samantha V Adams, Charlie Kirkwood, J Scott Hosking, Niall H Robinson, Sebastian Lerch, and Aneesh C Subramanian. Towards implementing artificial intelligence post-processing in weather and climate: proposed actions from the Oxford 2019 workshop. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200091, 2021.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020a.

Sijie He, Xinyan Li, Timothy DelSole, Pradeep Ravikumar, and Arindam Banerjee. Sub-seasonal climate forecasting via machine learning: Challenges, analysis, and advances, 2020b.

Sijie He, Xinyan Li, Laurie Trenary, Benjamin A Cash, Timothy DelSole, and Arindam Banerjee. Learning and dynamical models for sub-seasonal climate forecasting: Comparison and collaboration, 2021.

Warren J Hehre, Robert F Stewart, and John A Pople. Self-consistent molecular-orbital methods. I. use of gaussian expansions of slater-type atomic orbitals. *The Journal of Chemical Physics*, 51(6):2657–2664, 1969.

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

Trygve Helgaker, Poul Jorgensen, and Jeppe Olsen. *Molecular electronic-structure theory*. John Wiley & Sons, 2013.

Gregory R Herman and Russ S Schumacher. "Dendrology" in numerical weather prediction: What random forests and logistic regression tell us about forecasting extreme precipitation. *Monthly Weather Review*, 146(6):1785–1812, 2018.

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

Jan Hermann, James Spencer, Kenny Choo, Antonio Mezzacapo, W Matthew C Foulkes, David Pfau, Giuseppe Carleo, and Frank Noé. Ab initio quantum chemistry with neural-network wavefunctions. *Nature Reviews Chemistry*, 7(10):692–709, 2023.

Florian Hess, Zahra Monfared, Manuel Brenner, and Daniel Durstewitz. Generalized Teacher Forcing for Learning Chaotic Dynamics. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 13017–13049. PMLR, 2023. URL `https://proceedings.mlr.press/v202/hess23a.html`.

Pradeep Hewage, Marcello Trovati, Ella Pereira, and Ardhendu Behera. Deep learning-based effective fine-grained weather forecasting model. *Pattern Analysis and Applications*, 24(1):343–366, 2021.

RW Higgins, A Leetmaa, Y Xue, and A Barnston. Dominant factors influencing the seasonal predictability of us precipitation and surface air temperature. *Journal of Climate*, 13(22):3994–4017, 2000.

Eckhard Hitzer. Introduction to Clifford's geometric algebra. *Journal of the Society of Instrument and Control Engineers*, 51(4):338–350, 2012.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Chin-Wei Huang, Jae Hyun Lim, and Aaron C Courville. A variational perspective on diffusion-based generative models and score matching. *Advances in Neural Information Processing Systems*, 34:22863–22876, 2021.

Michael F. Hutchinson. 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.

Marcus Hutter. On representing (anti) symmetric functions. *arXiv preprint arXiv:2007.15298*, 2020.

Jessica Hwang, Paulo Orenstein, Karl Pfeiffer, Judah Cohen, and Lester Mackey. Improving subseasonal forecasting in the western US with machine learning, 2018.

Jessica Hwang, Paulo Orenstein, Judah Cohen, Karl Pfeiffer, and Lester Mackey. Improving subseasonal forecasting in the western US with machine learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2325–2335, 2019.

James M Hyman and Basil Nicolaenko. The kuramoto-sivashinsky equation: a bridge between pde's and dynamical systems. *Physica D: Nonlinear Phenomena*, 18(1-3):113–126, 1986.

Gilberto Iglesias, David C Kale, and Yan Liu. An examination of deep learning for extreme climate pattern analysis. In *The 5th International Workshop on Climate Informatics*, 2015.

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

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Hicham Janati, Marco Cuturi, and Alexandre Gramfort. Debiased Sinkhorn barycenters. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4692–4701, Online, 13–18 Jul 2020. PMLR. URL `https://proceedings.mlr.press/v119/janati20a.html`.

Robert Jastrow. Many-body problem with strong forces. *Physical Review*, 98(5):1479, 1955.

Ruoxi Jiang and Rebecca Willett. Embed and emulate: Learning to estimate parameters of dynamical systems with uncertainty quantification. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=6scShPCpdDu`.

Ruoxi Jiang, Peter Y Lu, Elena Orlova, and Rebecca Willett. Training neural operators to preserve invariant measures of chaotic attractors. *Advances in Neural Information Processing Systems*, 36:27645–27669, 2023.

Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. *IEEE transactions on visualization and computer graphics*, 26 (11):3365–3385, 2019.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 694–711. Springer, 2016.

Eugenia Kalnay, Masao Kanamitsu, Robert Kistler, William Collins, Dennis Deaven, Lev Gandin, Mark Iredell, Suranjana Saha, Glenn White, John Woollen, et al. The NCEP/NCAR 40-year reanalysis project. *Bulletin of the American meteorological Society*, 77(3):437–472, 1996.

Tosio Kato. On the eigenfunctions of many-particle systems in quantum mechanics. *Communications on Pure and Applied Mathematics*, 10(2):151–177, 1957.

John A Keith, Valentin Vassilev-Galindo, Bingqing Cheng, Stefan Chmiela, Michael Gastegger, Klaus-Robert Muller, and Alexandre Tkatchenko. Combining machine learning and computational chemistry for predictive insights into chemical systems. *Chemical reviews*, 121(16):9816–9872, 2021.

Abdelwahed Khamis, Russell Tsuchida, Mohamed Tarek, Vivien Rolland, and Lars Petersson. Earth movers in the big data era: A review of optimal transport in machine learning, 2023.

Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022.

Yuehaw Khoo, Jianfeng Lu, and Lexing Ying. Solving parametric pde problems with artificial neural networks. *European Journal of Applied Mathematics*, 32(3):421–435, 2021.

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

Matthieu Kirchmeyer, Yuan Yin, Jérémie Donà, Nicolas Baskiotis, Alain Rakotomamonjy, and Patrick Gallinari. Generalizing to new physical systems via context-informed dynamics model. In *International Conference on Machine Learning*, pages 11283–11301. PMLR, 2022.

Ben P Kirtman, Dughong Min, Johnna M Infanti, James L Kinter, Daniel A Paolino, Qin Zhang, Huug Van Den Dool, Suranjana Saha, Malaquias Pena Mendez, Emily Becker, et al. The North American multimodel ensemble: phase-1 seasonal-to-interannual prediction; phase-2 toward developing intraseasonal prediction. *Bulletin of the American Meteorological Society*, 95(4):585–601, 2014.

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

Anthony W Knapp and Anthony William Knapp. *Lie groups beyond an introduction*, volume 140. Springer, 1996.

Kenneth R Knapp, Steve Ansari, Caroline L Bain, Mark A Bourassa, Michael J Dickinson, Chris Funk, Chip N Helms, Christopher C Hennon, Christopher D Holmes, George J Huffman, et al. Globally gridded satellite observations for climate studies. *Bulletin of the American Meteorological Society*, 92(7):893–907, 2011.

Peter J Knowles and Nicholas C Handy. A new determinant-based full configuration interaction method. *Chemical physics letters*, 111(4-5):315–321, 1984.

Walter Kohn and L Sham. Density functional theory. In *Conference Proceedings-Italian Physical Society*, volume 49, pages 561–572. Editrice Compositori, 1996.

Patrick T. Komiske, Eric M. Metodiev, and Jesse Thaler. Metric space of collider events. *Phys. Rev. Lett.*, 123:041801, Jul 2019. doi:10.1103/PhysRevLett.123.041801. URL `https://link.aps.org/doi/10.1103/PhysRevLett.123.041801`.

Pascal Krause, Tillmann Klamroth, and Peter Saalfrank. Time-dependent configuration-interaction calculations of laser-pulse-driven many-electron dynamics: Controlled dipole switching in lithium cyanide. *The Journal of chemical physics*, 123(7), 2005.

Pascal Krause, Tillmann Klamroth, and Peter Saalfrank. Molecular response properties from explicitly time-dependent configuration interaction methods. *The Journal of chemical physics*, 127(3), 2007.

E Krotscheck, G-X Qian, and W Kohn. Theory of inhomogeneous quantum systems. I. Static properties of Bose fluids. *Physical Review B*, 31(7):4245, 1985.

Stanislaw A Kucharski and Rodney J Bartlett. The coupled-cluster single, double, triple, and quadruple excitation method. *The Journal of chemical physics*, 97(6):4282–4288, 1992.

Folkert Kuipers. *Stochastic mechanics: The unification of quantum mechanics with Brownian motion*. Springer Nature, 2023.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Learning skillful medium-range global weather forecasting. *Science*, page eadi2336, 2023.

Kody JH Law, Daniel Sanz-Alonso, Abhishek Shukla, and Andrew M Stuart. Filter accuracy for the Lorenz 96 model: Fixed versus adaptive observation operators. *Physica D: Nonlinear Phenomena*, 325:1–13, 2016.

Qianxiao Li, Felix Dietrich, Erik M Bollt, and Ioannis G Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27 (10):103111, 2017.

Xiaosong Li, Stanley M Smith, Alexei N Markevitch, Dmitri A Romanov, Robert J Levis, and H Bernhard Schlegel. A time-dependent Hartree–Fock approach for studying the electronic optical response of molecules in intense fields. *Physical Chemistry Chemical Physics*, 7(2):233–239, 2005.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *arXiv preprint arXiv:2205.13671*, 2022a.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Zongyi Li, Miguel Liu-Schiaffini, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Learning chaotic dynamics in dissipative systems. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022b. URL https://openreview.net/forum?id=1C36tFZn7sR.

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

Hongbin Liu, Andrew J Jenkins, Andrew Wildman, Michael J Frisch, Filippo Lipparini, Benedetta Mennucci, and Xiaosong Li. Time-dependent complete active space embedded in a polarizable force field. *Journal of chemical theory and computation*, 15(3):1633–1641, 2019.

Rong-Xiang Liu, Bo Tian, Li-Cai Liu, Bo Qin, and Xing Lü. 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.

Axel UJ Lode, Camille Lévêque, Lars Bojer Madsen, Alexej I Streltsov, and Ofir E Alon. Colloquium: Multiconfigurational time-dependent Hartree approaches for indistinguishable particles. *Reviews of Modern Physics*, 92(1):011001, 2020.

Eric D. Loken, Adam J. Clark, and Amy McGovern. Comparing and interpreting differently designed random forests for next-day severe weather hazard prediction. *Weather and Forecasting*, 37(6):871 – 899, 2022. doi:https://doi.org/10.1175/WAF-D-21-0138.1. URL https://journals.ametsoc.org/view/journals/wefo/37/6/WAF-D-21-0138.1.xml.

Andrew C Lorenc. Analysis methods for numerical weather prediction. *Quarterly Journal of the Royal Meteorological Society*, 112(474):1177–1194, 1986.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *arXiv preprint arXiv:2206.00927*, 2022a.

Fei Lu, Kevin K Lin, and Alexandre J Chorin. Data-based stochastic model reduction for the kuramoto–sivashinsky equation. *Physica D: Nonlinear Phenomena*, 340:46–57, 2017.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, Mar 2021. ISSN 2522-5839. doi:10.1038/s42256-021-00302-5.

Lu Lu, Xuhui Meng, Shengze Cai, Zhiping Mao, Somdatta Goswami, Zhongqiang Zhang, and George Em Karniadakis. A comprehensive and fair comparison of two neural operators (with practical extensions) based on fair data. *Computer Methods in Applied Mechanics and Engineering*, 393:114778, 2022b.

Peter Y. Lu, Samuel Kim, and Marin Soljačić. Extracting interpretable physical parameters from spatiotemporal systems using unsupervised learning. *Phys. Rev. X*, 10:031056, Sep 2020. doi:10.1103/PhysRevX.10.031056. URL https://link.aps.org/doi/10.1103/PhysRevX.10.031056.

Peter Y. Lu, Rumen Dangovski, and Marin Soljačić. Discovering conservation laws using optimal transport and manifold learning. *Nat. Commun.*, 14(1):4744, 7 August 2023. URL http://dx.doi.org/10.1038/s41467-023-40325-7.

Hongjun Luo and Ali Alavi. Combining the transcorrelated method with full configuration interaction quantum Monte Carlo: Application to the homogeneous electron gas. *Journal of chemical theory and computation*, 14(3):1403–1411, 2018.

Eleonora Luppi and Martin Head-Gordon. Computation of high-harmonic generation spectra of H2 and N2 in intense laser pulses using quantum chemistry methods and time-dependent density functional theory. *Molecular Physics*, 110(9-10):909–923, 2012.

Vamshi C Madala, Shivkumar Chandrasekaran, and Jason Bunk. CNNs avoid curse of dimensionality by learning on patches. *IEEE Open Journal of Signal Processing*, 2023.

Andrew J Majda and John Harlim. *Filtering Complex Turbulent Systems*. Cambridge University Press, 2012.

Andrew J Majda, John Harlim, and Boris Gershgorin. Mathematical strategies for filtering turbulent dynamical systems. *Discrete & Continuous Dynamical Systems*, 27(2):441, 2010.

Antonios Mamalakis, Jin-Yi Yu, James T Randerson, Amir AghaKouchak, and Efi Foufoula-Georgiou. A new interhemispheric teleconnection increases predictability of winter precipitation in southwestern us. *Nature communications*, 9(1):2332, 2018.

Antonios Mamalakis, Amir AghaKouchak, James T. Randerson, and Efi Foufoula-Georgiou. Hotspots of predictability: Identifying regions of high precipitation predictability at seasonal timescales from limited time series observations. *Water Resources Research*, 58 (5):e2021WR031302, 2022. doi:https://doi.org/10.1029/2021WR031302. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2021WR031302. e2021WR031302 2021WR031302.

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

A. Mathews, M. Francisquez, J. W. Hughes, D. R. Hatch, B. Zhu, and B. N. Rogers. Uncovering turbulent plasma dynamics via deep learning from partial observations. *Phys. Rev. E*, 104:025205, Aug 2021. doi:10.1103/PhysRevE.104.025205. URL https://link.aps.org/doi/10.1103/PhysRevE.104.025205.

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

AD McLachlan. Van der waals forces between an atom and a surface. *Molecular Physics*, 7 (4):381–388, 1964.

AD McLachlan and MA Ball. Time-dependent Hartree—Fock theory for molecules. *Reviews of Modern Physics*, 36(3):844, 1964.

Alfredo Medio and Marji Lines. *Nonlinear Dynamics: A Primer*. Cambridge University Press, Cambridge, 2001. doi:10.1017/CBO9780511754050.

Nicolai Meinshausen. Quantile regression forests. *Journal of Machine Learning Research*, 7 (35):983–999, 2006. URL http://jmlr.org/papers/v7/meinshausen06a.html.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Jonas M Mikhaeil, Zahra Monfared, and Daniel Durstewitz. On the difficulty of learning chaotic dynamics with RNNs. In S Koyejo, S Mohamed, A Agarwal, D Belgrave, K Cho, and A Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 11297–11312. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/495e55f361708bedbab5d81f92048dcd-Paper-Conference.pdf.

Young-Mi Min, Suryun Ham, Jin-Ho Yoo, and Su-Hee Han. Recent progress and future prospects of subseasonal and seasonal climate predictions. *Bulletin of the American Meteorological Society*, 101(5):E640–E644, 2020.

Soukayna Mouatadid, Paulo Orenstein, Genevieve Flaspohler, Judah Cohen, Miruna Oprescu, Ernest Fraenkel, and Lester Mackey. Adaptive bias correction for improved subseasonal forecasting. *Nature Communications*, 14(1):3482, 2023.

Albert Musaelian, Simon Batzner, Anders Johansson, Lixin Sun, Cameron J. Owen, Mordechai Kornbluth, and Boris Kozinsky. Learning local equivariant representations for large-scale atomistic dynamics. *Nature Communications*, 14(1):579, Feb 2023. ISSN 2041-1723. doi:10.1038/s41467-023-36329-y.

Boris Muzellec, Kanji Sato, Mathurin Massias, and Taiji Suzuki. Dimension-free convergence rates for gradient Langevin dynamics in RKHS, 2020.

R Nagaraj and Lakshmi Sutha Kumar. Univariate deep learning models for prediction of daily average temperature and relative humidity: The case study of chennai, india. *Journal of Earth System Science*, 132(3):100, 2023.

Kazumi Nakada, Robin M Kovach, Jelena Marshak, and Andrea Molod. Global modeling and assimilation office - NASA, Apr 2018. URL `https://gmao.gsfc.nasa.gov/pubs/docs/Nakada1033.pdf`.

Hiroshi Nakatsuji. 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.

Annamalai Narayanan, Mahinthan Chandramohan, Lihui Chen, Yang Liu, and Santhoshkumar Saminathan. subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs, 2016.

Daniel R Nascimento and A Eugene DePrince III. Linear absorption spectra from explicitly time-dependent equation-of-motion coupled-cluster theory. *Journal of chemical theory and computation*, 12(12):5834–5840, 2016.

National Academies of Sciences. *Next generation earth system prediction: strategies for subseasonal to seasonal forecasts*. National Academies Press, 2016.

National Research Council. *Assessment of intraseasonal to interannual climate prediction and predictability*. National Academies Press, 2010.

Frederik Nebeker. *Calculating the weather: Meteorology in the 20th century*, volume 60. Elsevier, 1995.

Kirill Neklyudov, Jannes Nys, Luca Thiede, Juan Carrasquilla, Qiang Liu, Max Welling, and Alireza Makhzani. Wasserstein quantum Monte Carlo: a novel approach for solving the quantum many-body Schrödinger equation. *Advances in Neural Information Processing Systems*, 36, 2024.

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

Edward Nelson. Derivation of the schrödinger equation from newtonian mechanics. *Physical review*, 150(4):1079, 1966b.

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

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

Edward Nelson. *Quantum fluctuations*, volume 16. Princeton University Press, 2020b.

Matthew Newman, Michael A Alexander, Toby R Ault, Kim M Cobb, Clara Deser, Emanuele Di Lorenzo, Nathan J Mantua, Arthur J Miller, Shoshiro Minobe, Hisashi Nakamura, et al. The pacific decadal oscillation, revisited. *Journal of Climate*, 29(12):4399–4427, 2016.

NOAA. NOAA National Centers for Environmental information, Climate at a Glance: National Time Series. https://www.ncdc.noaa.gov/cag/, 2022.

NOAA National Centers for Environmental Prediction, Climate Prediction Center. Official 30-day forecasts of precipitation and temperature over united states. https://www.cpc.ncep.noaa.gov/products/predictions/30day/, 2019.

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

Nikolas Nüsken and Lorenz Richter. 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.

Jannes Nys, Gabriel Pescia, Alessandro Sinibaldi, and Giuseppe Carleo. Ab-initio variational wave functions for the time-dependent many-electron schrödinger equation. *Nature Communications*, 15(1):9404, 2024.

Elena Orlova, Aleksei Ustimenko, Ruoxi Jiang, Peter Y Lu, and Rebecca Willett. Deep stochastic mechanics. *arXiv preprint arXiv:2305.19685*, 2023.

Elena Orlova, Haokun Liu, Raphael Rossellini, Benjamin A Cash, and Rebecca Willett. Beyond ensemble averages: Leveraging climate model ensembles for subseasonal forecasting. *Artificial Intelligence for the Earth Systems*, 3(4):e230103, 2024a.

Elena Orlova, Aleksei Ustimenko, Ruoxi Jiang, Peter Y. Lu, and Rebecca Willett. Deep stochastic mechanics. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 38779–38814. PMLR, 21–27 Jul 2024b. URL https://proceedings.mlr.press/v235/orlova24a.html.

Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33 (5):2295–2317, 2011.

Baoxiang Pan, Kuolin Hsu, Amir AghaKouchak, Soroosh Sorooshian, and Wayne Higgins. Precipitation prediction skill for the west coast united states: From short to extended range. *Journal of Climate*, 32(1):161–182, 2019.

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

Omiros Papaspiliopoulos and Gareth Roberts. Importance sampling techniques for estimation of diffusion models. *Statistical methods for stochastic differential equations*, 124: 311–340, 2012.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch, 2017.

Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al. FourCastNet: A global data-driven high-resolution weather model using adaptive Fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

Bohdan Pavlyshenko. Using stacking approaches for machine learning models. In *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*, pages 255–258. IEEE, 2018.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases?, 2019.

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

David Pfau, James S. Spencer, Alexander G. D. G. Matthews, and W. M. C. Foulkes. Ab initio solution of the many-electron schrödinger equation with deep neural networks. *Phys. Rev. Res.*, 2:033429, Sep 2020b. doi:10.1103/PhysRevResearch.2.033429. URL `https://link.aps.org/doi/10.1103/PhysRevResearch.2.033429`.

Thomas Pichler, Marcello Dalmonte, Enrique Rico, Peter Zoller, and Simone Montangero. Real-time dynamics in u (1) lattice gauge theories with tensor networks. *Physical Review X*, 6(1):011023, 2016.

Jason A Platt, Stephen G Penny, Timothy A Smith, Tse-Chun Chen, and Henry D I Abarbanel. Constraining chaos: Enforcing dynamical invariants in the training of reservoir computers. *Chaos*, 33(10), 1 October 2023. URL `http://dx.doi.org/10.1063/5.0156999`.

Thomas Plé, Louis Lagardère, and Jean-Philip Piquemal. Force-field-enhanced neural network interactions: from local equivariant embedding to atom-in-molecule properties and long-range effects. *Chemical Science*, 14(44):12554–12569, 2023.

Tomaso Poggio, Hrushikesh Mhaskar, Lorenzo Rosasco, Brando Miranda, and Qianli Liao. 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.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Timo Ewalds, Andrew El-Kadi, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, Remi Lam, and Matthew Willson. GenCast: Diffusion-based ensemble forecasting for medium-range weather. *arXiv preprint arXiv:2312.15796*, 2023.

Carlos Tejero Prieto and Raffaele Vitolo. 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`.

Y Radhika and M Shashi. Atmospheric temperature prediction using support vector machines. *International journal of computer theory and engineering*, 1(1):55, 2009.

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

Md Ashiqur Rahman, Manuel A Florez, Anima Anandkumar, Zachary E Ross, and Kamyar Azizzadenesheli. Generative adversarial neural operators. *arXiv preprint arXiv:2205.03017*, 2022.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. 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.

Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. Equivariance through parameter-sharing. In *International conference on machine learning*, pages 2892–2901. PMLR, 2017.

Mohsen Razavy. *Quantum theory of tunneling*. World Scientific, 2013.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of ADAM and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

Markus Reichstein, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, and F Prabhat. Deep learning and process understanding for data-driven earth system science. *Nature*, 566(7743):195–204, 2019.

Richard W Reynolds, Thomas M Smith, Chunying Liu, Dudley B Chelton, Kenneth S Casey, and Michael G Schlax. Daily high-resolution-blended analyses for sea surface temperature. *Journal of climate*, 20(22):5473–5496, 2007.

Yaniv Romano, Evan Patterson, and Emmanuel J. Candès. Conformalized quantile regression. In *NeurIPS*, 2019.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *MICCAI*, 2015.

Ribana Roscher, Bastian Bohn, Marco F Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *Ieee Access*, 8:42200–42216, 2020.

Suranjana Saha, Shrinivas Moorthi, Xingren Wu, Jiande Wang, Sudhir Nadiga, Patrick Tripp, David Behringer, Yu-Tai Hou, Hui-ya Chuang, Mark Iredell, et al. The NCEP climate forecast system version 2. *Journal of climate*, 27(6):2185–2208, 2014.

Cristopher Salvi, Maud Lemercier, and Andris Gerasimovics. Neural stochastic pdes: Resolution-invariant learning of continuous spatiotemporal dynamics. *arXiv preprint arXiv:2110.10249*, 2021.

Takeshi Sato and Kenichi L Ishikawa. Time-dependent complete-active-space self-consistent-field method for multielectron dynamics in intense laser fields. *Physical Review A—Atomic, Molecular, and Optical Physics*, 88(2):023402, 2013.

Michael Scherbela, Rafael Reisenhofer, Leon Gerard, Philipp Marquetand, and Philipp Grohs. Solving the electronic schrödinger equation for multiple nuclear geometries with weight-sharing deep neural networks. *Nature Computational Science*, 2(5):331–341, 2022.

Michael Scherbela, Leon Gerard, and Philipp Grohs. Towards a foundation model for neural network wavefunctions. *arXiv preprint arXiv:2303.09949*, 2023.

H Bernhard Schlegel, Stanley M Smith, and Xiaosong Li. Electronic optical response of molecules in intense fields: Comparison of TD-HF, TD-CIS, and TD-CIS (D) approaches. *The Journal of chemical physics*, 126(24), 2007.

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

Peter J Schmid. Dynamic mode decomposition and its variants. *Annual Review of Fluid Mechanics*, 54:225–254, 2022.

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

Kristof T Schütt, Michael Gastegger, Alexandre Tkatchenko, K-R Müller, and Reinhard J Maurer. Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions. *Nature communications*, 10(1):5024, 2019.

Richard Seager, Mingfang Ting, Isaac Held, Yochanan Kushnir, Jian Lu, Gabriel Vecchi, Huei-Ping Huang, Nili Harnik, Ants Leetmaa, Ngar-Cheung Lau, et al. Model projections of an imminent transition to a more arid climate in southwestern north america. *Science*, 316(5828):1181–1184, 2007.

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

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

Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. doi:10.1002/j.1538-7305.1948.tb01338.x.

Isaiah Shavitt. The method of configuration interaction. In *Methods of electronic structure theory*, pages 189–275. Springer, 1977.

Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, oct 1997. doi:10.1137/s0097539795293172. URL `https://doi.org/10.1137%2Fs0097539795293172`.

Ryan Simeon. Intoroduction to Clifford algebras and used in representation theory. *The University of Chicago Department of Mathematics*, 2019. URL `https://math.uchicago.edu/~may/REU2019/REUPapers/Simeon.pdf`.

A. J. Simmons and A. Hollingsworth. Some aspects of the improvement in skill of numerical weather prediction. *Quarterly Journal of the Royal Meteorological Society*, 128(580):647–677, 2002.

Alessandro Sinibaldi, Clemens Giuliani, Giuseppe Carleo, and Filippo Vicentini. Unbiasing time-dependent Variational Monte Carlo by projected quantum evolution. *arXiv preprint arXiv:2305.14294*, 2023.

Andreas S Skeidsvoll, Alice Balbi, and Henrik Koch. Time-dependent coupled-cluster theory for ultrafast transient-absorption spectroscopy. *Physical Review A*, 102(2):023115, 2020.

John C Slater. Atomic shielding constants. *Physical review*, 36(1):57, 1930.

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

Vishwak Srinivasan, Justin Khim, Arindam Banerjee, and Pradeep Ravikumar. Subseasonal climate prediction in the Western US using bayesian spatial models. In *Uncertainty in artificial intelligence*, pages 961–970. PMLR, 2021.

James Stokes, Brian Chen, and Shravan Veerapaneni. Numerical and geometrical aspects of flow-based variational quantum Monte Carlo. *Machine Learning: Science and Technology*, 4(2):021001, 2023.

Adarsh Subbaswamy, Roy Adams, and Suchi Saria. Evaluating model robustness and stability to dataset shift. In *International Conference on Artificial Intelligence and Statistics*, pages 2611–2619. PMLR, 2021.

Qiming Sun, Timothy C Berkelbach, Nick S Blunt, George H Booth, Sheng Guo, Zhendong Li, Junzi Liu, James D McClain, Elvira R Sayfutyarova, Sandeep Sharma, et al. Pyscf: the Python-based simulations of chemistry framework. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 8(1):e1340, 2018.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 1596–1611. Curran Associates, Inc., 2022.

James D Talman. Linked-cluster expansion for Jastrow-type wave functions and its application to the electron-gas problem. *Physical Review A*, 10(4):1333, 1974.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*, 2019.

Yunjie Tian, Lingxi Xie, Xiaopeng Zhang, Jiemin Fang, Haohang Xu, Wei Huang, Jianbin Jiao, Qi Tian, and Qixiang Ye. Semantic-aware generation for self-supervised visual representation learning. *arXiv preprint arXiv:2111.13163*, 2021.

Sonja Totz, Eli Tziperman, Dim Coumou, Karl Pfeiffer, and Judah Cohen. Winter precipitation forecast in the European and Mediterranean regions using cluster analysis. *Geophysical Research Letters*, 44(24):12–418, 2017.

Matthias Troyer and Uwe-Jens Wiese. Computational complexity and fundamental limitations to fermionic Quantum Monte Carlo simulations. *Physical review letters*, 94(17): 170201, 2005.

Carsten A. Ullrich. *Time-Dependent Density-Functional Theory: Concepts and Applications.* Oxford University Press, 12 2011. ISBN 9780199563029. doi:10.1093/acprof:oso/9780199563029.001.0001. URL https://doi.org/10.1093/acprof:oso/9780199563029.001.0001.

Oliver T Unke and Markus Meuwly. Physnet: A neural network for predicting energies, forces, dipole moments, and partial charges. *Journal of chemical theory and computation*, 15(6):3678–3693, 2019.

Oliver T. Unke, Stefan Chmiela, Huziel E. Sauceda, Michael Gastegger, Igor Poltavsky, Kristof T. Schütt, Alexandre Tkatchenko, and Klaus-Robert Müller. Machine learning force fields. *Chemical Reviews*, 121(16):10142–10186, Aug 2021. ISSN 0009-2665. doi:10.1021/acs.chemrev.0c01111.

Shashank Reddy Vadyala and Sai Nethra Betgeri. General implementation of quantum physics-informed neural network. *Array*, page 100287, 2023.

Dirk Leendert van Kekem. Dynamics of the lorenz-96 model: Bifurcations, symmetries and waves, 2018. https://research.rug.nl/en/publications/dynamics-of-the-lorenz-96-model-bifurcations-symmetries-and-waves.

S Vannitsem, JB Bremnes, J Demaeyer, GR Evans, J Flowerdew, S Hemri, S Lerch, N Roberts, S Theis, A Atencia, et al. Statistical postprocessing for weather forecasts: Review, challenges, and avenues in a big data world. *Bulletin of the American Meteorological Society*, 102(3):E681–E699, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Filippo Vicentini, Damian Hofmann, Attila Szabó, Dian Wu, Christopher Roth, Clemens Giuliani, Gabriel Pescia, Jannes Nys, Vladimir Vargas-Calderón, Nikita Astrakhantsev, et al. NetKet 3: Machine learning toolbox for many-body quantum systems. *SciPost Physics Codebases*, page 007, 2022.

Cédric Villani. *Optimal Transport: Old and New.* Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-71050-9. doi:10.1007/978-3-540-71050-9_6.

Ė.B. Vinberg. *A Course in Algebra.* Graduate studies in mathematics. American Mathematical Society, 2003. ISBN 9780821834138. URL https://books.google.com/books?id=kd24d3mwaecC.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17 (3):261–272, 2020.

Frédéric Vitart, Andrew W Robertson, and David LT Anderson. Subseasonal to seasonal prediction project: Bridging the gap between weather and climate. *Bulletin of the World Meteorological Organization*, 61(2):23, 2012.

Pantelis R. Vlachas, Georgios Arampatzis, Caroline Uhler, and Petros Koumoutsakos. Multiscale simulations of complex systems by learning their effective dynamics, 2021.

Pantelis R Vlachas, Georgios Arampatzis, Caroline Uhler, and Petros Koumoutsakos. Multiscale simulations of complex systems by learning their effective dynamics. *Nature Machine Intelligence*, 4(4):359–366, 2022.

Ingrid von Glehn, James S Spencer, and David Pfau. A self-attention ansatz for ab-initio quantum chemistry. *arXiv preprint arXiv:2211.13672*, 2022.

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

Timothy C Wallstrom. The stochastic mechanics of the Pauli equation. *Transactions of the American Mathematical Society*, 318(2):749–762, 1990.

Chuanfu Wang, Chunlei Fan, and Qun Ding. Constructing discrete chaotic systems with positive lyapunov exponents. *International Journal of Bifurcation and Chaos*, 28(07): 1850084, 2018.

Rui Wang, Robin Walters, and Rose Yu. Meta-learning dynamics forecasting using task inference. *Advances in Neural Information Processing Systems*, 35:21640–21653, 2022a.

Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why PINNs fail to train: a neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022b.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

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

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

Christopher J White, Daniela IV Domeisen, Nachiketa Acharya, Elijah A Adefisan, Michael L Anderson, Stella Aura, Ahmed A Balogun, Douglas Bertram, Sonia Bluhm, David J Brayshaw, et al. Advances in the application and utility of subseasonal-to-seasonal predictions. *Bulletin of the American Meteorological Society*, 103(6):E1448–E1472, 2022.

Olivia Wiles, Sven Gowal, Florian Stimberg, Sylvestre Alvise-Rebuffi, Ira Ktena, Taylan Cemgil, et al. A fine-grained analysis on distribution shift, 2021.

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

Aleksander P Woźniak, Michał Przybytek, Maciej Lewenstein, and Robert Moszyński. Effects of electronic correlation on the high harmonic generation in helium: A time-dependent configuration interaction singles vs time-dependent full configuration interaction study. *The Journal of Chemical Physics*, 156(17), 2022.

Kan Wu, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. Rethinking and improving relative position encoding for vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10033–10041, 2021.

Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Robert E Wyatt. *Quantum dynamics with trajectories: introduction to quantum hydrodynamics*, volume 28. Springer Science & Business Media, 2005.

P Xie, M Chen, and W Shi. Cpc global unified gauge-based analysis of daily precipitation. In *Preprints, 24th Conf. on Hydrology, Atlanta, GA, Amer. Metero. Soc*, volume 2, 2010.

Kazuhiro Yabana and GF Bertsch. Time-dependent local-density approximation in real time. *Physical Review B*, 54(7):4484, 1996.

Pavel Yakubovskiy. Segmentation models pytorch. `https://github.com/qubvel/segmentation_models.pytorch`, 2020.

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

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796*, 2022.

Shuxian Yang, Fenghua Ling, Yue Li, and Jing-Jia Luo. Improving seasonal prediction of summer precipitation in the middle–lower reaches of the yangtze river using a tu-net deep learning approach. *Artificial Intelligence for the Earth Systems*, 2(2):220078, 2023a.

Yunan Yang, Levon Nurbekyan, Elisa Negrini, Robert Martin, and Mirjeta Pasha. Optimal transport for parameter identification of chaotic dynamics via invariant measures. *SIAM Journal on Applied Dynamical Systems*, 22(1):269–310, 2023b. doi:10.1137/21M1421337.

Yong-Xin Yao, Niladri Gomes, Feng Zhang, Cai-Zhuang Wang, Kai-Ming Ho, Thomas Iadecola, and Peter P Orth. Adaptive variational quantum dynamics simulations. *PRX Quantum*, 2(3):030307, 2021.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

Xiao Zhang and Michael Maire. Self-supervised visual representation learning from hierarchical grouping. *Advances in Neural Information Processing Systems*, 33:16579–16590, 2020.

Bowen Zhu, Jingwei Hu, Yifei Lou, and Yunan Yang. Implicit regularization effects of the sobolev norms in image processing. *arXiv preprint arXiv:2109.06255*, 2021a.

Qi Zhu, Natalia Ponomareva, Jiawei Han, and Bryan Perozzi. Shift-robust gnns: Overcoming the limitations of localized graph training data. *Advances in Neural Information Processing Systems*, 34:27965–27977, 2021b.

Aaron Zweig and Joan Bruna. Towards antisymmetric neural ansatz separation. *arXiv preprint arXiv:2208.03264*, 2022.

APPENDIX A

ADDITIONAL RESULTS FOR SUBSEASONAL FORECASTING

A.1 Regression results for for NASA-GMAO dataset

A.1.1 Regression

Precipitation regression using NASA-GMAO Precipitation regression results on the test data from NASA-GMAO are presented in Table A.1. On this dataset, no learned method or method leveraging ensemble model forecasts significantly outperforms climatology. Note that the best R^2 value associated with the climatology is still negative; the low values for all methods indicate the difficulty of the forecasting problem.

Table A.1: Test results for precipitation regression using NASA-GMAO dataset. LR refers to linear regression on all features including ensemble members, lagged data, land variables, and SSTs. Model stacking is performed on models that are learned on all features. Bold values indicate the best performance for each statistic. MSE is reported in squared mm.

Model	Features	Mean R^2 (↑)	Median R^2 (↑)	Mean Sq Err (↓)	Median MSE (↓)	90th prctl MSE (↓)
Baseline	Climatology	**-0.07**	**-0.02**	2.14 ± 0.04	**1.51**	4.40
	Ens mean	-0.11	-0.06	2.13 ± 0.04	1.52	4.31
	Linear Regr	-0.18	-0.14	2.25 ± 0.04	1.62	4.68
LR	All features	-0.40	-0.29	2.62 ± 0.05	1.93	5.42
U-Net	All features	-0.19	-0.09	2.11 ± 0.03	1.56	**4.25**
RF	All features	-0.18	-0.11	2.17 ± 0.04	1.55	4.44
Stacked	LR, U-Net, RF, outputs	-0.08	-0.06	**2.09 ± 0.04**	1.52	4.27

Figure A.1 illustrates the test performance of key methods on NASA-GMAO data with R^2 heatmaps over the U.S. Although the stacked model does not show the best performance in terms of mean R^2 score, it has more geographic regions with positive R^2 than any other method.

Regression of temperature Temperature regression results using NASA-GMAO ensemble members are presented in Table A.2. The random forest and linear regression outperform

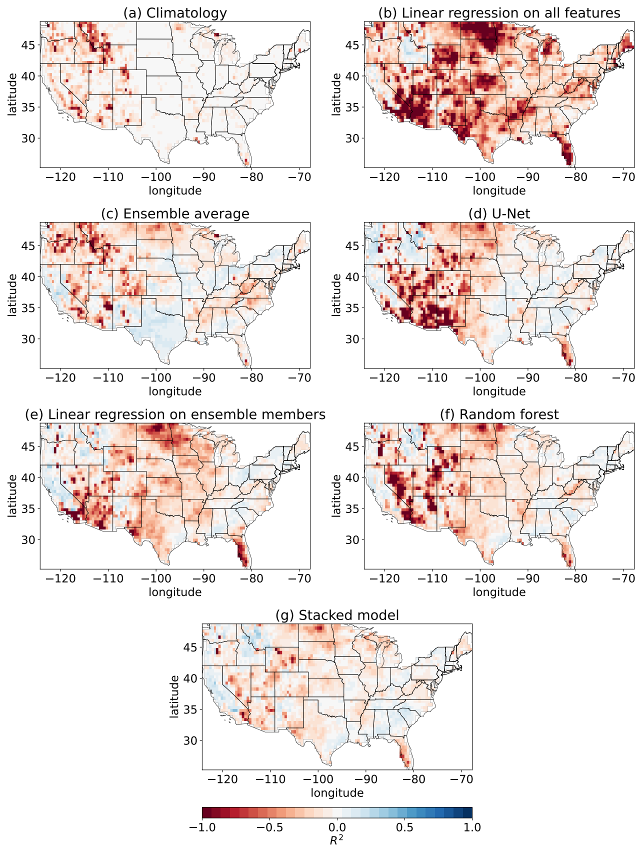


Figure A.1: Test R^2 score heatmaps of baselines and learning-based methods for precipitation regression using the NASA-GMAO dataset. Positive values (blue) indicate better performance. See Appendix A.1.1 for details.

all baselines in terms of both R^2 score and MSE. However, the U-Net model's performance is lower compared to other learned methods, which might be a sign of overfitting. Despite this performance drop of U-Net, the model stacking approach still demonstrates the best predictive skill. Note that the model stacking approach is applied to the models that are trained on all available features except SSTs (similar to NCEP-CFSv2 data).

Figure A.2 illustrates the test performance of key methods on NASA-GMAO data with R^2 heatmaps over the U.S. The stacked model shows the best performance across spatial locations. Similar to the NCEP-CFSv2 dataset, we notice that there are still regions where all models tend to exhibit a negative R^2 score.

Table A.2: Test results for temperature regression using NASA-GMAO dataset. LR refers to linear regression on all features including ensemble members, lagged data, and land variables. Model stacking is performed on models that are learned on all features except SSTs. Bold values indicate the best performance for each statistic. MSE is reported in squared $^\circ C$.

Model	Features	Mean R^2 (↑)	Median R^2 (↑)	Mean Sq Err (↓)	Median MSE (↓)	90th prctl MSE (↓)
Baseline	Climatology	-0.70	-0.20	6.49 ± 0.11	5.06	9.72
	Ens mean	-0.28	0.12	4.82 ± 0.10	3.43	7.82
	Linear Regr	0.12	0.14	3.32 ± 0.02	3.11	4.70
LR	All features wo SSTs	0.17	0.17	3.10± 0.02	3.05	4.26
U-Net	All features wo SSTs	0.06	0.12	3.40 ± 0.02	3.27	4.52
RF	All features wo SSTs	0.20	0.22	3.03 ± 0.02	2.94	4.25
Stacked	LR, U-Net, RF, outputs	**0.21**	**0.22**	**2.94 ± 0.02**	**2.89**	**3.97**

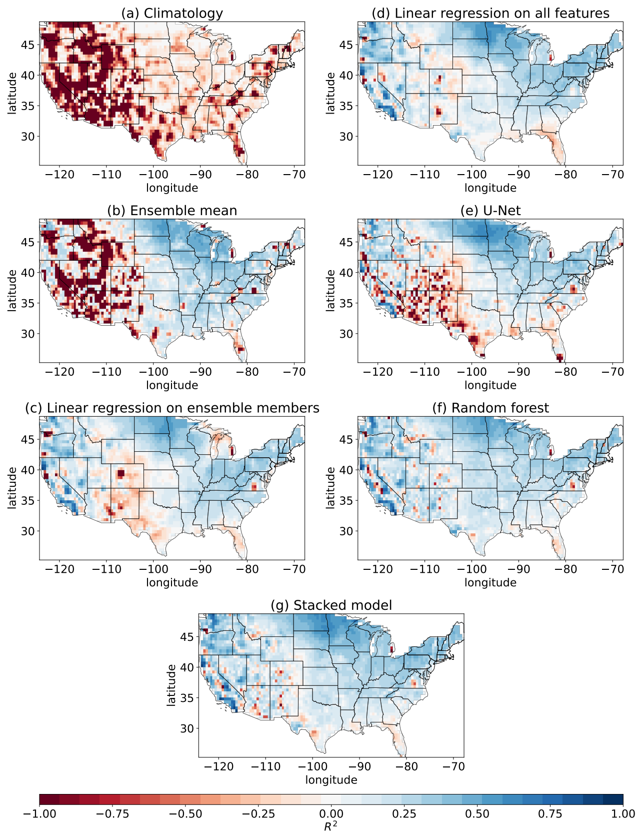


Figure A.2: Test R^2 score heatmaps of baselines and learning-based methods for temperature regression using NASA-GMAO dataset. Positive values (blue) indicate better performance. See Appendix A.1.1 for details.

A.1.2 Quantile regression

Quantile regression of precipitation using NCEP-CFSv2 ensemble In Figure A.3, we show heatmaps of quantile loss using all locations in the U.S., where blue means smaller quantile loss and yellow means larger quantile loss. We observe that the learning-based models outperform the baselines, especially in Washington, California, Idaho, and near the Gulf of Mexico.

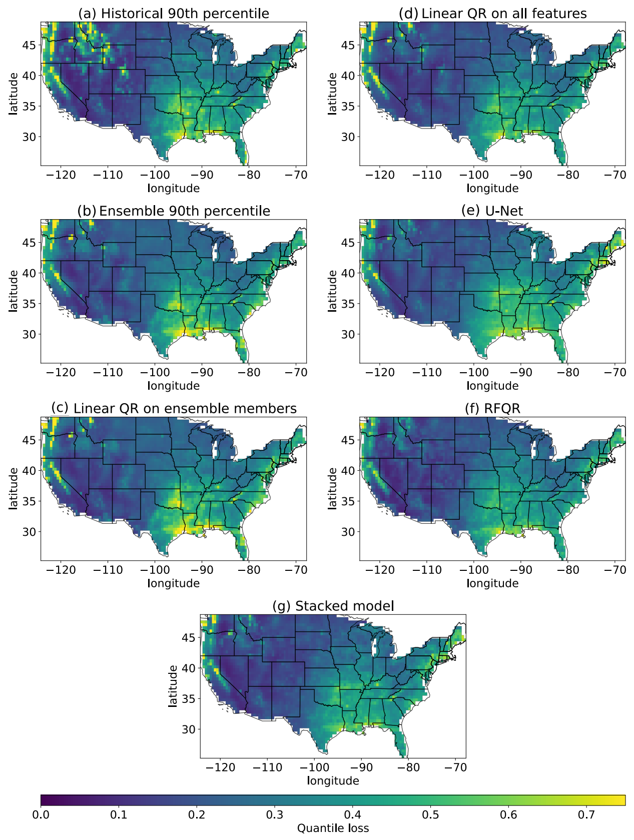

(a) Historical 90th percentile
(d) Linear QR on all features
(b) Ensemble 90th percentile
(e) U-Net
(c) Linear QR on ensemble members
(f) RFQR
(g) Stacked model
latitude
longitude
45
40
35
30
−120
−110
−100
−90
−80
−70
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Quantile loss

Figure A.3: Test quantile loss heatmaps of baselines and learning-based methods for precipitation quantile regression using NCEP-CFSv2 dataset. Blue regions indicate smaller quantile loss. See Section 2.6.2 for details.

Quantile regression of precipitation using NASA-GMAO ensemble Table A.3 summarizes results for precipitation quantile regression using the NASA-GMAO ensemble. The models are the same as the models applied to the NCEP-CFSv2 ensemble. Our best model shows similar performance to that of the baselines. One possible reason is that the NASA-GMAO ensemble shows worse performance than the NCEP-CFSv2 ensemble empirically. Furthermore, according to the designs of ensemble members from both climate models, the NASA-GMAO ensemble has fewer ensemble members and, therefore, has less coverage on the distribution of precipitation than the NCEP-CFSv2 ensemble, so our learned model has access to less information about the true distribution of precipitation.

Table A.3: Test results for precipitation quantile regression using NASA-GMAO dataset, with target quantile = 0.9. Model stacking is performed on models that are learned on all features. The **best** results are in bold. Quantile loss is reported in mm.

Model	Features	Mean Qtr Loss (↓)	Median Qtr Loss (↓)	90th prctl Qtr Loss (↓)
Baseline	Historical 90-th percentile	0.295 ± 0.003	0.263	0.484
	Ens 90-th percentile	0.378± 0.005	0.308	0.673
	Linear QR ens only	0.336 ± 0.004	0.286	0.531
Linear QR	All features	**0.290 ± 0.003**	**0.253**	**0.456**
U-Net	All features	0.310 ± 0.002	0.278	0.489
RFQR	All features	0.290 ± 0.002	0.265	0.471
Stacked	U-Net, RFQR, LQR outputs	0.296± 0.002	0.268	0.467

Quantile regression of temperature Table A.4 and Figure A.5 summarize results for temperature quantile regression using the NASA-GMAO ensemble. All of our learned models are able to outperform all baselines.

Table A.4: Test results for temperature quantile regression using NASA-GMAO dataset, with target quantile = 0.9. Model stacking is performed on models that are learned on all features except for SSTs. The **best** results are in bold. Quantile loss is reported in $^{\circ}C$.

Model	Features	Mean Qtr Loss (↓)	Median Qtr Loss (↓)	90th prctl Qtr Loss (↓)
Baseline	Historical 90-th percentile	0.596 ± 0.010	0.438	0.988
	Ens 90-th percentile	0.812 ± 0.009	0.646	1.493
	Linear QR ens only	0.445 ± 0.003	0.411	0.618
Linear QR	All features wo SSTs	0.341 ± 0.001	0.333	0.419
U-Net	All features wo SSTs	0.375 ± 0.003	0.347	0.477
RFQR	All features wo SSTs	0.318 ± 0.002	0.316	0.376
Stacked	U-Net, RFQR, LQR outputs	**0.315** ± 0.002	**0.310**	**0.374**

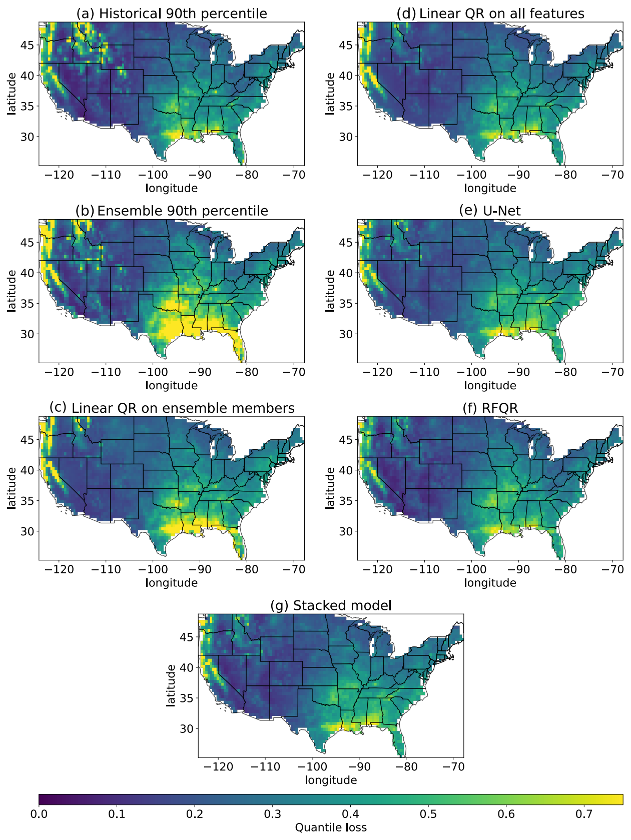

(a) Historical 90th percentile
(d) Linear QR on all features
(b) Ensemble 90th percentile
(e) U-Net
(c) Linear QR on ensemble members
(f) RFQR
(g) Stacked model
latitude
longitude
45
40
35
30
−120
−110
−100
−90
−80
−70
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Quantile loss

Figure A.4: Test quantile loss heatmaps of baselines and learning-based methods for precipitation quantile regression using NASA-GMAO dataset. Blue regions indicate smaller quantile loss. See Appendix A.1.2 for details.

A.2 Tercile Classification

In this section, we present results for the tercile classification task for both climate variables and both datasets.

A.2.1 Tercile classification of precipitation

In this case, the proposed learning-based methods are directly trained on the classification task. Predictions of baselines, such as the ensemble mean, are split into three classes according to the 33rd and 66th percentile values. Note that random forest and U-Net are trained for classification using all available features. We do not notice a significant difference in

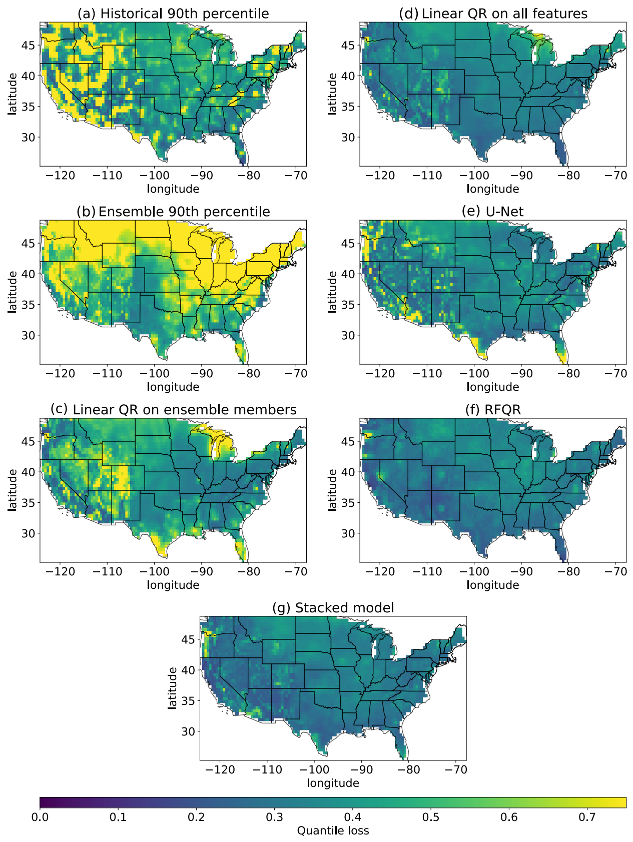

(a) Historical 90th percentile
(b) Ensemble 90th percentile
(c) Linear QR on ensemble members
(d) Linear QR on all features
(e) U-Net
(f) RFQR
(g) Stacked model
latitude
longitude
Quantile loss

Figure A.5: Test quantile loss heatmaps of baselines and learning-based methods for temperature quantile regression using NASA-GMAO dataset. Blue regions indicate smaller quantile loss. See Appendix A.1.2 for details.

the performance of logistic regression on the validation if the inputs are ensemble members only or ensemble members with side information. So, we use logistic regression on ensemble members only. The model stacking is applied to the logistic regression, U-Net, and random forest outputs.

Table A.5 summarizes results for NCEP-CFSv2 and NASA-GMAO datasets on the test data. For this task, the learning-based methods achieve the best performance in terms of accuracy for both datasets. In the case of NCEP-CFSv2 data, U-Net achieves the highest accuracy score, and the performance of the stacked model is comparable with it. For NASA-GMAO data, the stacked model shows the best performance.

The accuracy heatmaps over U. S. land are presented in the Figure A.6 for NCEP-CFSv2

Table A.5: Test results for tercile classification of precipitation on different datasets. Accuracy in % is reported. Note that for this task, our models are trained for classification directly while baselines perform regression, and a threshold for predicted values is applied. For stacking, logistic regression, U-Net and RF outputs are used.

Data	Model	Mean accuracy (↑)	Median accuracy (↑)
NCEP-CFSv2	Ens mean	38.00 ±0.16	37.61
	Logistic Regr	41.22 ±0.14	40.17
	U-Net	**43.88** ±0.12	**42.74**
	RF	42.38 ±0.13	41.88
	Stacked	43.81 ±0.13	**42.74**
NASA-GMAO	Ens mean	38.64 ±0.14	37.65
	Logistic regr	41.51 ±0.16	40.00
	U-Net	40.53 ±0.11	40.00
	RF	40.79 ±0.14	40.00
	Stacked	**42.08** ±0.14	**41.18**

dataset. The plots corresponding to the learning-based methods show the best results, especially at the West Coast, Colorado and North America.

The accuracy heatmaps over the U. S. land are presented in Figure A.7 for the NASA-GMAO dataset. The plots corresponding to the learning-based methods show the best results, the ensemble mean's figure has the most red regions.

A.2.2 Tercile classification of temperature

The next task is tercile classification of 2-meter temperature. In this case, the threshold is applied to the regression predictions of all methods, meaning there is no direct training for a classification. Table A.6 summarizes results for NCEP-CFSv2 and NASA-GMAO datasets on the test data. For this task, the learning-based methods achieve the best performance in terms of accuracy, stacked model using NCEP-CFSv2 data and linear regression using NASA-GMAO data and all additional features (except SSTs). In general, all learning-based models significantly outperform the ensemble mean.

Figure A.8 shows accuracy heatmaps over the U.S. for different methods using NCEP-CFSv2 data. The stacked model shows the best performance across spatial locations. For

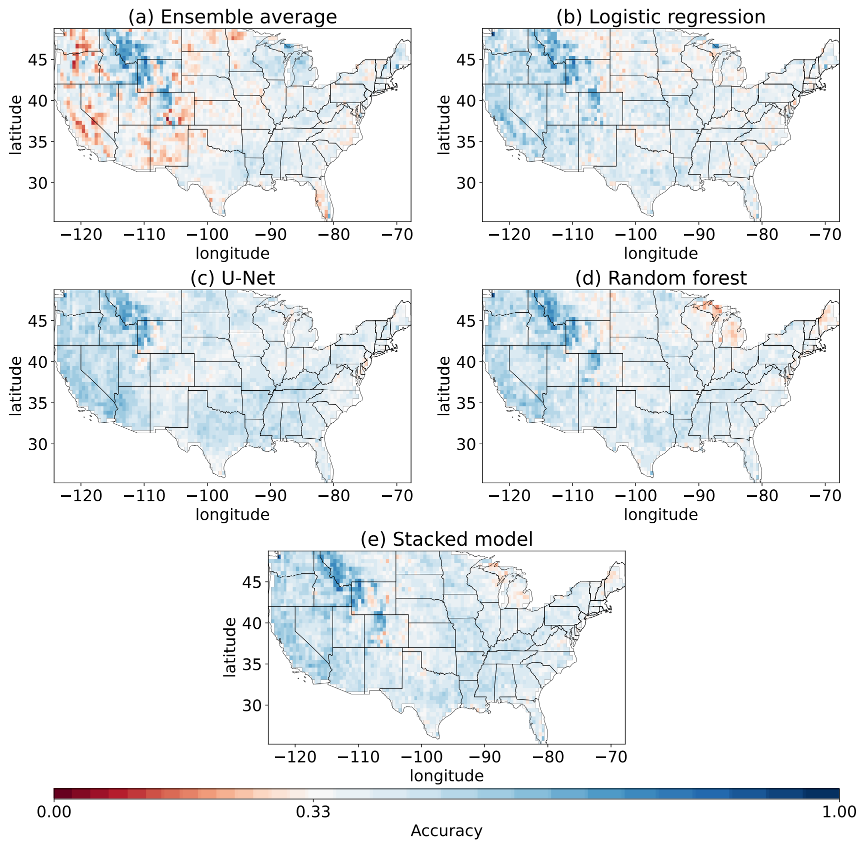


Figure A.6: Test accuracy heatmaps of baselines and learning-based methods for tercile classification of precipitation using NCEP-CFSv2 dataset. The accuracy colorbar is recentered to be white at $\frac{1}{3}$, what corresponds to a random guess score. Blue pixels indicate better performance, while red pixels correspond to performance that is worse than a random guess. See Appendix A.2.1 for details.

example, the ensemble mean does not show great performance in the Southeast and Middle Atlantic regions, while learning-based methods demonstrate much stronger predictive skills in these areas. However, there are still some areas, such as Texas or South West region, with red pixels for all methods.

Figure A.9 shows accuracy heatmaps over the U.S. for different methods using NASA-GMAO data. In this case, linear regression on all features achieves the best scores. Other learning-based methods outperform the ensemble mean too, especially in the West and in Minnesota.

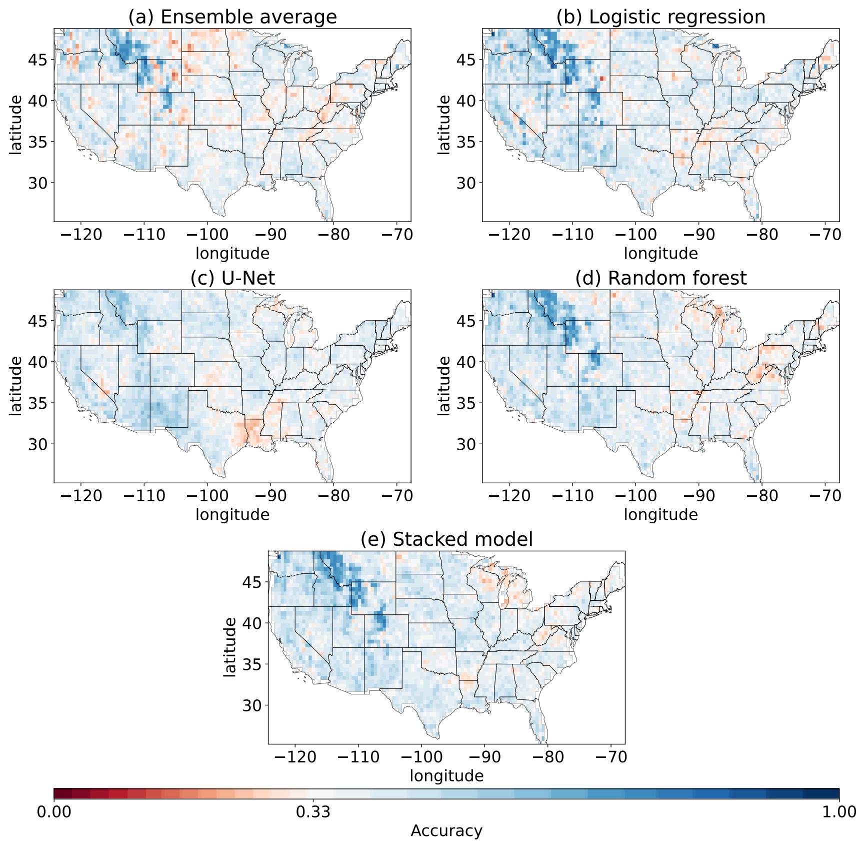


Figure A.7: Test accuracy heatmaps of baselines and learning-based methods for tercile classification of precipitation using NASA-GMAO dataset. The accuracy colorbar is recentered to be white at $\frac{1}{3}$, what corresponds to a random guess score. Blue pixels indicate better performance, while red pixels correspond to performance that is worse than a random guess. See Appendix A.2.1 for details.

A.3 Extended Discussions

In this section, we present more detailed results for Section 2.7. We also provide additional experiments on the temperature forecasting analysis, and experiments on training set sizes and bootstrap.

A.3.1 PE v.s. latitude/longitude values or no location information

In this section, we elaborate on our experiment with different uses of location information. Similar to Section 2.7.2, we train and test our models with three different settings: using

Table A.6: Test results for tercile classification of temperature on different datasets. Accuracy in % is reported. Note that for this task, our models are trained for regression and the threshold for predicted values is applied.

Data	Model	Mean accuracy (↑)	Median accuracy (↑)
NCEP-CFSv2	Ens mean	44.84 ±0.36	42.74
	Linear Regr	57.10 ±0.25	54.69
	LR	57.34 ±0.25	**54.71**
	U-Net	53.80 ±0.28	50.43
	RF	58.07 ±0.27	54.70
	Stacked	**58.12** ±0.20	**54.71**
NASA-GMAO	Ens mean	52.23 ±0.25	49.41
	Linear Regr	57.75 ±0.25	54.11
	LR	**58.97** ±0.25	**55.29**
	U-Net	55.64 ±0.27	51.76
	RF	58.78 ±0.26	**55.29**
	Stacked	58.72 ±0.25	54.12

no location information, using latitude/longitude values, or using positional encodings. For the stacked model, we first train the LR, RF, and U-Net using these different settings before training the stacked model using the corresponding LR, RF, and U-Net outputs. Table A.7 summarizes test performance for precipitation regression with these three settings of using location information. For linear regression, we observe that having latitude/longitude values or adding PE features does not improve its performance. One interesting result is that adding PE for the LR degraded its performance, which may be due to the fact that the PE features are non-linear transformations of latitude and longitude values, which is hard to fit with a linear model.

For the RF, U-Net, and the stacked model, we observe that adding PE improves their performance with significance, i.e. having at least one standard error smaller MSE. For the U-Net, using latitude/longitude values yields worse overall performance compared to using no location information. Figure A.10 shows the test R^2 heatmaps for LR, U-Net, RF, and stacked model under these three settings for precipitation regression. We observe that adding PE features not only improves performance for the U-Net and RF, the stacked

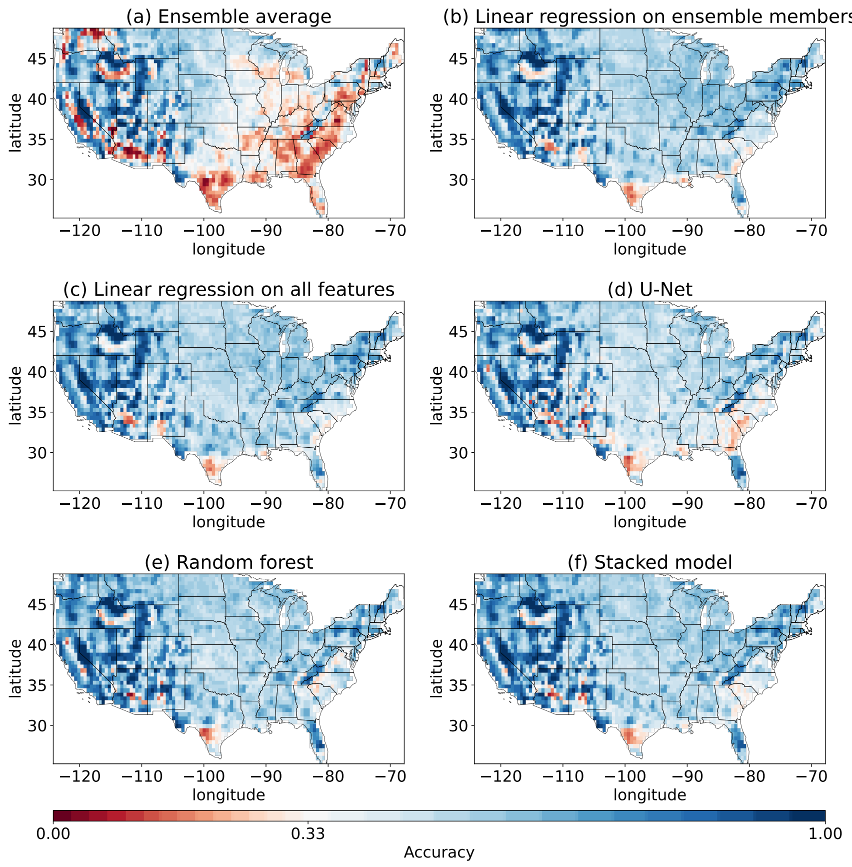


Figure A.8: Test accuracy heatmaps of baselines and learning-based methods for tercile classification of temperature using NCEP-CFSv2 dataset. The accuracy colorbar is recentered to be white at $\frac{1}{3}$, what corresponds to a random guess score. Blue pixels indicate better performance, while red pixels correspond to performance that is worse than a random guess. See Appendix A.2.2 for details.

model's performance also improves from having better predictions from the U-Net and RF.

Table A.8 shows the test performances of LR, U-Net, RF and stacked model on temperature regression. Similar to precipitation regression, adding latitude/longitude values or adding PE does not help the LR, but we observe significant performance improvement when adding positional encoding features for the U-Net, RF, and the stacked model. Figure A.11 then shows the test R^2 heatmaps for temperature regression. We can see that adding PE to the U-Net, RF, and stacked model improves forecast performance, especially in regions like Arizona, New Mexico, and Texas.

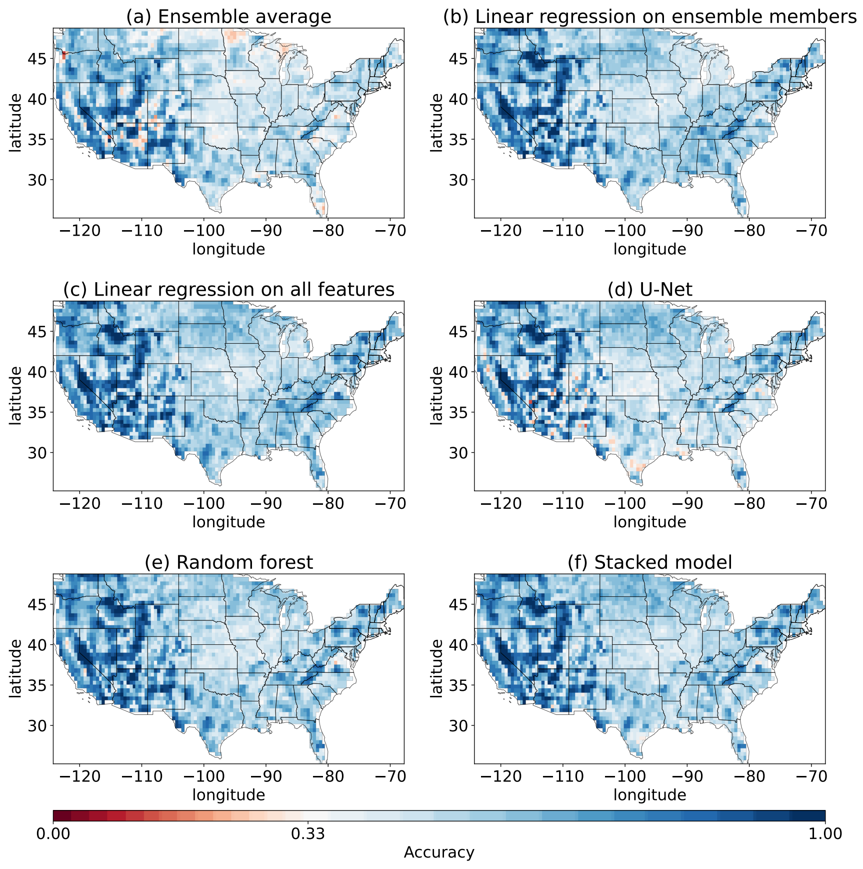


Figure A.9: Test accuracy heatmaps of baselines and learning-based methods for tercile classification of temperature using NASA-GMAO dataset. The accuracy colorbar is recentered to be white at $\frac{1}{3}$, what corresponds to a random guess score. Blue pixels indicate better performance, while red pixels correspond to performance that is worse than a random guess. See Appendix A.2.2 for details.

A.3.2 Bootstrap experiments

To evaluate the stability of our machine learning models with small sample sizes, we perform the following bootstrap experiments: We take bootstrap samples of size 200 from our training set and retrain our U-Net, RF, and LR. Then we evaluate these models on the test set. We repeat this process 50 times and show the results in Figure A.12. We observe from the plots that the U-Net performs consistently better than the LR in precipitation regression but not for temperature regression. This result is consistent with what we showed in Table 2.3 and Table 2.4.

Table A.7: Precipitation regression test performance comparison of LR, U-Net, RF and stacked model trained using no spatial features, using latitude and longitude values or using PE. The **best** results are in bold.

Model	Features	Mean R^2 (↑)	Mean Sq Err (↓)
LR	All + no location info	**-0.11**	**2.29**±0.04
	All + lat/lon values	**-0.11**	**2.29**±0.04
	All + PE	-0.33	2.71±0.05
U-Net	All + no location info	-0.16	2.31±0.04
	All + lat/lon values	-0.28	2.53±0.05
	All + PE	**-0.10**	**2.18**± 0.03
RF	All + no location info	-0.18	2.23±0.04
	All + lat/lon values	-0.16	2.21±0.04
	All + PE	**-0.11**	**2.17**± 0.05
Stacked	All + no location info	-0.05	2.13±0.03
	All + lat/lon values	-0.01	2.21±0.04
	All + PE	**0.02**	**2.07**± 0.03

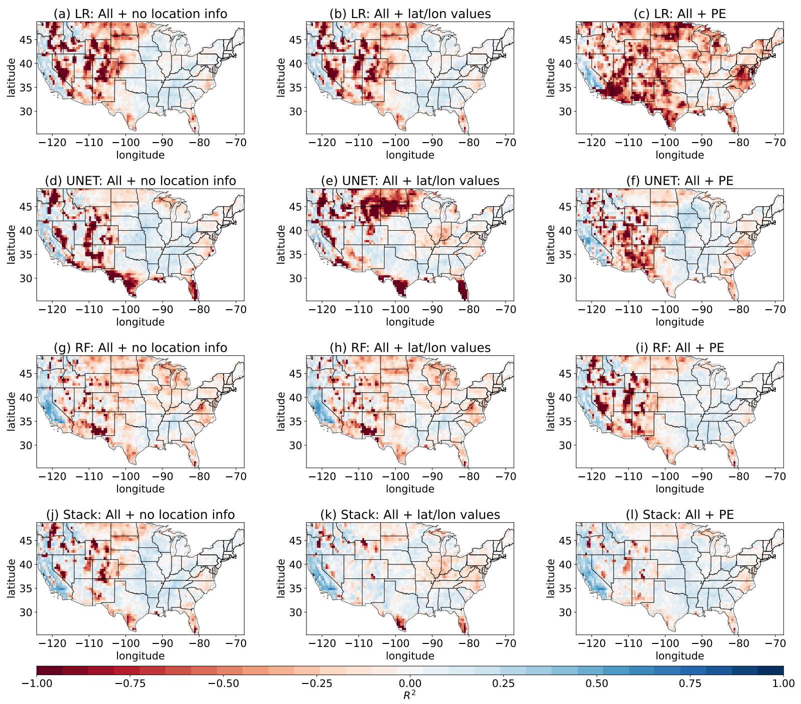

(a) LR: All + no location info
(b) LR: All + lat/lon values
(c) LR: All + PE
(d) UNET: All + no location info
(e) UNET: All + lat/lon values
(f) UNET: All + PE
(g) RF: All + no location info
(h) RF: All + lat/lon values
(i) RF: All + PE
(j) Stack: All + no location info
(k) Stack: All + lat/lon values
(l) Stack: All + PE
latitude
longitude
R^2

Figure A.10: Precipitation regression test R^2 heatmaps of LR, U-Net, RF and stacked model trained using no spatial features, using latitude and longitude values or using PE. The NCEP-CFSv2 ensemble is used. See Appendix A.3.1 for more details.

We also observe that the U-Net is more sensitive to different bootstrap samples than the RF and LR, which is not surprising since for the U-Net, the bootstrap samples correspond

Table A.8: Temperature regression test performance comparison of LR, U-Net, RF and stacked model trained using no spatial features, using latitude and longitude values or using PE. The **best** results are in bold.

Model	Features	Mean R^2 (↑)	Mean Sq Err (↓)
LR	All + no location info	**0.05**	**3.57**±0.03
	All + lat/lon values	**0.05**	**3.57**±0.03
	All + PE	**0.05**	3.57±0.03
U-Net	All + no location info	-0.35	4.81±0.04
	All + lat/lon values	-0.21	4.47±0.03
	All + PE	**0.01**	**3.65**± 0.02
RF	All + no location info	0.11	3.37±0.02
	All + lat/lon values	0.14	3.28±0.02
	All + PE	**0.16**	**3.17**± 0.02
Stacked	All + no location info	0.12	3.35±0.02
	All + lat/lon values	0.12	3.33±0.02
	All + PE	**0.18**	**3.11**± 0.02

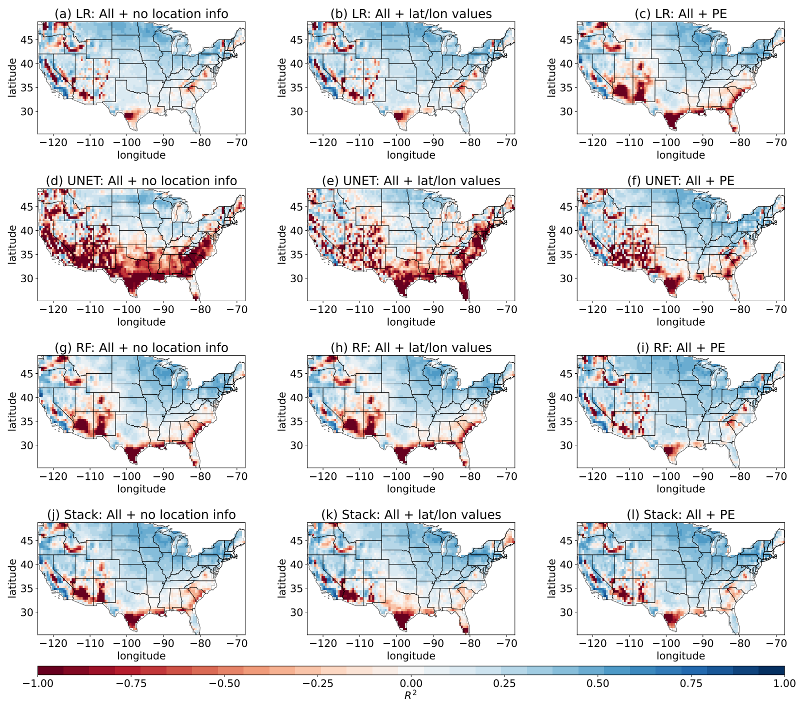

(a) LR: All + no location info
(b) LR: All + lat/lon values
(c) LR: All + PE
(d) UNET: All + no location info
(e) UNET: All + lat/lon values
(f) UNET: All + PE
(g) RF: All + no location info
(h) RF: All + lat/lon values
(i) RF: All + PE
(j) Stack: All + no location info
(k) Stack: All + lat/lon values
(l) Stack: All + PE
latitude
longitude
R^2

Figure A.11: Temperature regression test R^2 heatmaps of LR, U-Net, RF and stacked model trained using no spatial features, using latitude and longitude values or using PE. The NCEP-CFSv2 ensemble is used. See Appendix A.3.1 for more details.

to 200 different spatial maps for training. In contrast, for the RF and LR, the bootstrap samples correspond to 200 × 3274 training samples.

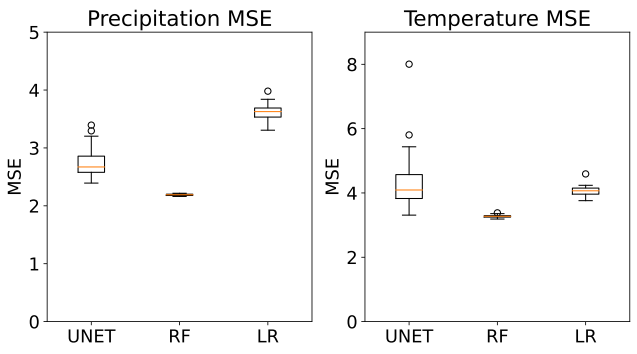
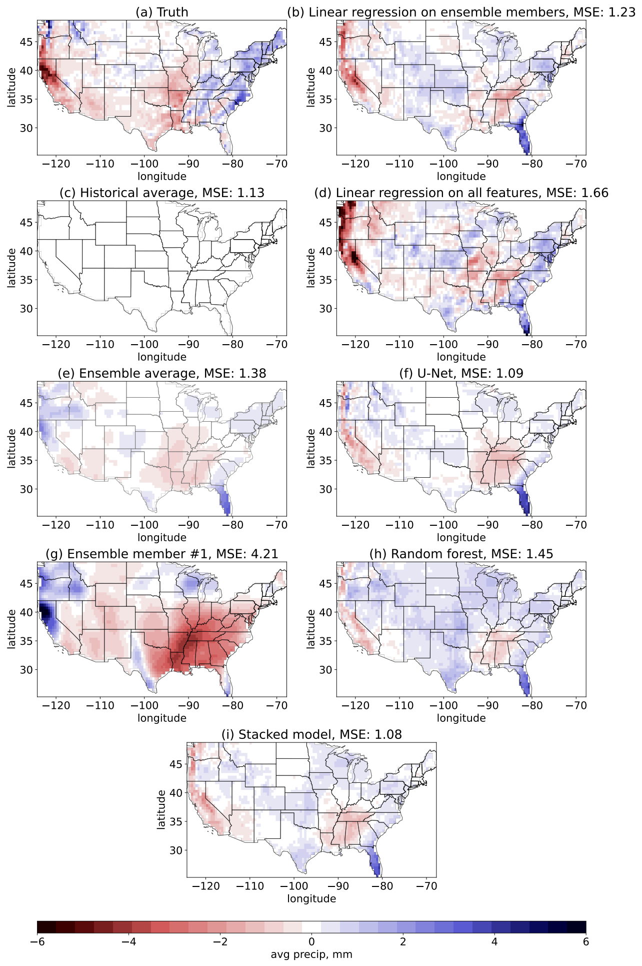


Figure A.12: Box plots of MSEs for the U-Net, LR and RF trained on 50 sets of different bootstrap samples, each with size 200. The NCEP-CFSv2 ensemble is used. See Appendix A.3.2 for more details.

A.3.3 Precipitation forecast example

While climate simulations and ensemble forecasts are designed to provide useful predictions of temperature and precipitation based on carefully developed physical models, we see that machine learning applied to those ensembles can yield a significantly higher predictive skill for a range of SSF tasks. Figure A.13 illustrates key differences between different predictive models for predicting monthly precipitation with a lead time of 14 days. Individual ensemble members are predictions with high levels of spatial smoothness and more extreme values. Linear regression, the random forest, the U-Net, and the stacked model produce higher spatial frequencies. The linear regression result, which uses a different model trained for each spatial location separately, has the least spatial smoothness of all methods; this is especially visible in the southeast and potentially does not reflect realistic spatial structure. The learning-based models more accurately predict localized regions of high and low precipitation compared to the ensemble mean.

Figure A.14 demonstrates differences between the ground truth and different model predictions. In this figure, the color white is associated with the smallest errors, while red pixels indicate overestimating precipitation and blue pixels indicate underestimating precipitation. The individual ensemble member in Figure A.14(e) exhibits dark red regions across



(a) Truth
(b) Linear regression on ensemble members, MSE: 1.23
(c) Historical average, MSE: 1.13
(d) Linear regression on all features, MSE: 1.66
(e) Ensemble average, MSE: 1.38
(f) U-Net, MSE: 1.09
(g) Ensemble member #1, MSE: 4.21
(h) Random forest, MSE: 1.45
(i) Stacked model, MSE: 1.08
latitude
longitude
avg precip, mm

Figure A.13: An illustration of precipitation predictions $\hat{y}_{t,l}^{\text{anomaly}}$ (in mm) of different methods for February 2016 (in test period). (a) True precipitation. (b) LR on ensemble members. (c) Climatology. (d) LR on all features. (e) Ensemble mean. (f) U-Net on all features. (g) Example single ensemble member. (h) Random forest on all features. (i) Stacked model. See Appendix A.3.3 for details.

the West, while the ensemble mean in Figure A.14(e) shows better performance in this area. The colors are more muted for the stacked model in Figure A.14(h). The climatology in Figure A.14(a) has the most neutral areas. However, its MSE is slightly higher than the stacked model's MSE. In general, all methods, including linear regression (b, d), U-Net (f), and random forest (g), tend to underpredict precipitation in the Southeast, Mid-Atlantic, and North Atlantic and predict higher precipitation levels in the West.

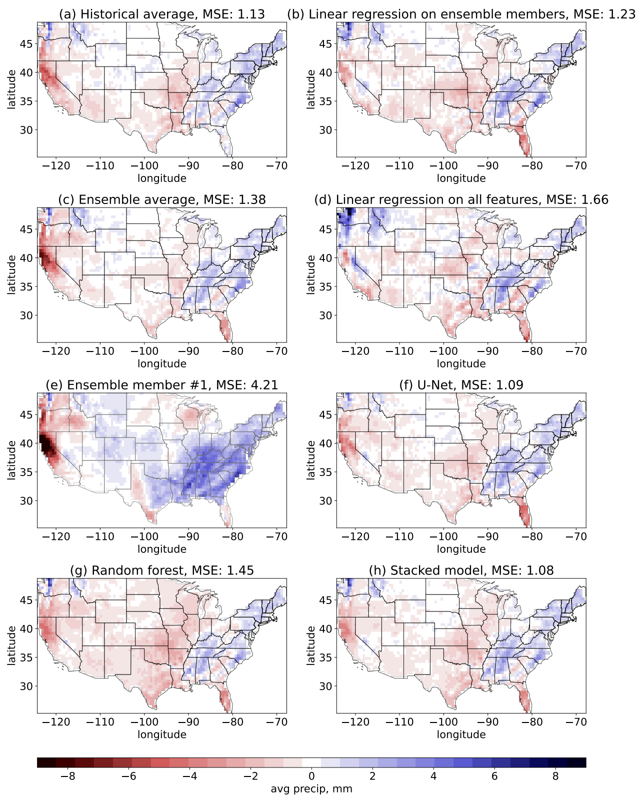

(a) Historical average, MSE: 1.13
(b) Linear regression on ensemble members, MSE: 1.23
(c) Ensemble average, MSE: 1.38
(d) Linear regression on all features, MSE: 1.66
(e) Ensemble member #1, MSE: 4.21
(f) U-Net, MSE: 1.09
(g) Random forest, MSE: 1.45
(h) Stacked model, MSE: 1.08
latitude
longitude
avg precip, mm

Figure A.14: An illustration of differences $y_{t,l}^{\text{anomaly}} - \hat{y}_{t,l}^{\text{anomaly}}$ in precipitation predictions in mm of different methods for February 2016 (in test period). Red pixels indicate areas where a forecasting method predicts higher precipitation levels compared to the ground truth, blue pixels indicate an underestimation of the precipitation, and white pixels correspond to a precise prediction. See Appendix A.3.3 for details.

A.3.4 Temperature forecasting analysis

Figure 2.3 shows regions in Texas and Florida where the ensemble mean and linear regression performance is poor, while a random forest achieves far superior performance. We conduct an analysis of forecasts of the ensemble mean, linear regression, and random forests in these regions together with a region in Wisconsin where all methods show good performance. Figure A.15 indicates these regions and Table A.9 summarizes the performance of different methods in these regions: the ensemble mean prediction quality dramatically drops between the validation and test periods in Texas and Florida, which is not the case for the random forest.

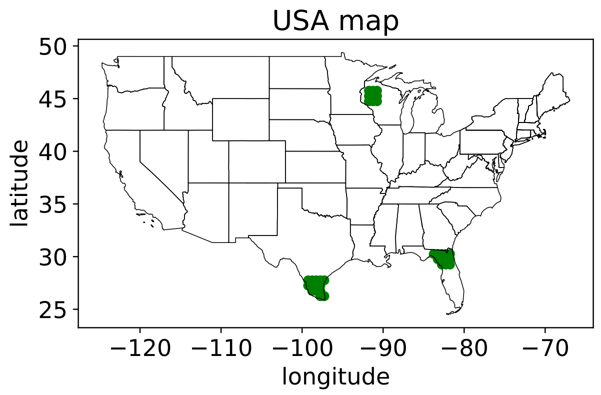


Figure A.15: Regions where the temperature forecast is analyzed. See Appendix A.3.4 for details.

Table A.9: Train, validation, and test performance of different methods in Texas, Florida, and Wisconsin regions. The task is temperature regression; NCEP-CFSv2 dataset is used. The performance of the ensemble mean and linear regression in the test period significantly decreases in Texas and Florida while the random forest is able to demonstrate reasonable results. All methods perform well in Wisconsin.

Region location	Model	Train mean R^2 ($\uparrow$)	Validation mean R^2 ($\uparrow$)	Test mean R^2 ($\uparrow$)
Texas	Ens mean	0.19	0.36	-1.55
	LR	0.53	0.49	-1.29
	RF	0.97	0.32	-0.33
Florida	Ens mean	0.11	0.34	-0.87
	LR	0.47	0.58	-0.56
	RF	0.97	0.36	0.11
Wisconsin	Ens mean	0.30	0.36	0.39
	LR	0.53	0.57	0.51
	RF	1.00	0.47	0.47

Why does RF perform so much better than simpler methods in some regions? One possibility is that the RF is a nonlinear model capable of more complex predictions. However, if that were the only cause of the discrepancy in performance, then we would expect that the RF would be better not only during the test period, but during the validation period as well. Table A.9 does not support this argument; it shows that the ensemble mean and linear regression have comparable, if not superior, performance to the random forest during the validation period. A second hypothesis is that the distribution of temperature is different during the test period than during the training and validation periods. This hypothesis is plausible for two reasons: (1) climate change, and (2) the training and validation data use hindcast ensembles while the test data uses forecast ensembles. To investigate this hypothesis, in Figure A.16 we plot the true temperature and ensemble mean in the training, validation, and test periods for the three geographic regions. The discrepancy between the true temperatures and ensemble means in the test period is generally greater than during the training and validation periods in Texas and Florida (though not in Wisconsin, a region where validation and test performance are comparable for all methods). This lends support to the hypothesis that hindcast and forecast ensembles exhibit distribution drift, and the superior performance of the RF during the test period may be due to a greater robustness to that distribution drift.

The hindcast and forecast ensembles may have different predictive accuracies because the hindcast ensembles have been debiased to fit past observations – a procedure not possible for forecast data. To explore the potential impact of debiasing, Figure A.16 shows the "oracle debiased ensemble mean", which is computed by using the test data to estimate the forecast ensemble bias and subtracting it from the ensemble mean. This procedure, *which would not be possible in practice and is used only to probe distribution drift ensemble bias*, yields smaller discrepancies between the true data and the (oracle debiased) ensemble mean than the discrepancies between the true data and the original (biased) ensemble mean. Specifically, the oracle ensemble member achieves -0.20 mean R^2 score (TX) and -0.28 mean R^2 score (FL) vs. -1.55 R^2 (TX) and -0.87 R^2 (FL) of the original forecast ensemble mean. The errors during the test period are generally larger than during the train and validation period, even after debiasing the ensemble members using future data. This effect may be attributed both to (a) the nonstationarity of the climate (note that there are more extreme values during the test period than during the training and validation periods, particularly in Texas and Florida) and (b) the fact that in the train and validation periods, we use hindcast ensemble members, whereas in the test period, we use forecast ensemble members.

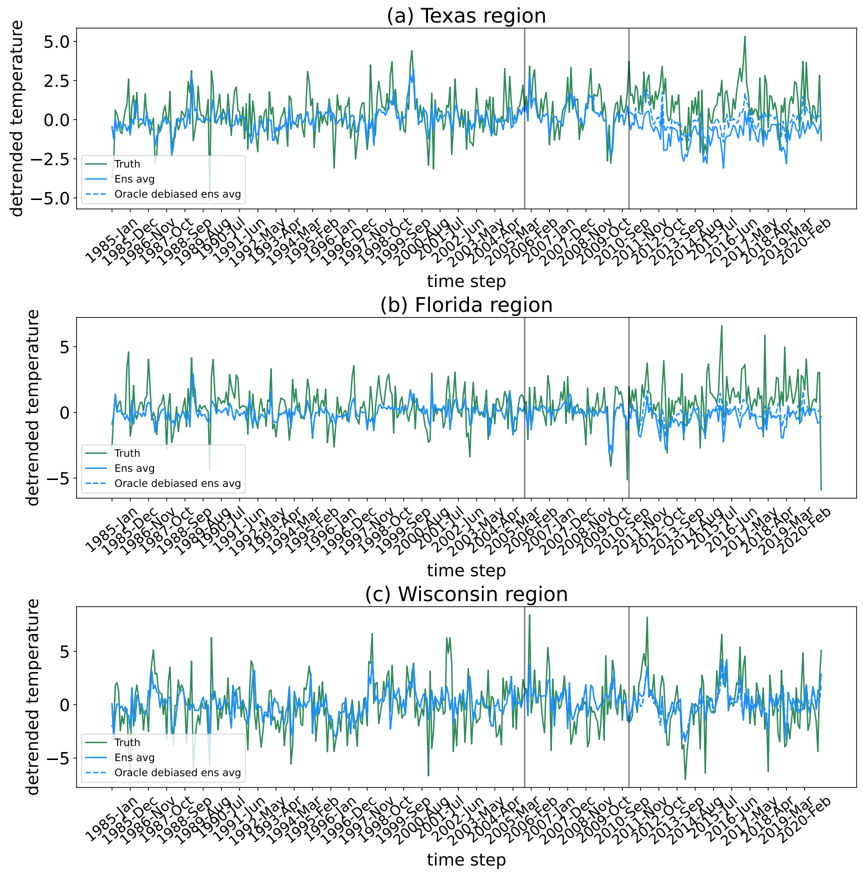


Figure A.16: Temperature predictions in $^{\circ}C$ of different methods at Texas, Florida, and Wisconsin regions. Black lines correspond to train/val and val/test splits; train and validation correspond to the hindcast regime of the ensemble, while test corresponds to the forecast regime. See Appendix A.3.4 for details.

A.4 Machine learning architectures

A.4.1 U-Net details

The U-Net has residual connections from layers in the encoder part to the decoder part in a paired way so that it forms a U-shape. Figure A.17 shows the architecture of the U-Net. The U-Net is a powerful deep convolutional network that is widely used in image processing tasks such as image segmentation [Ronneberger et al., 2015, Hao et al., 2020] or style transfer

[Gatys et al., 2016, Jing et al., 2019].

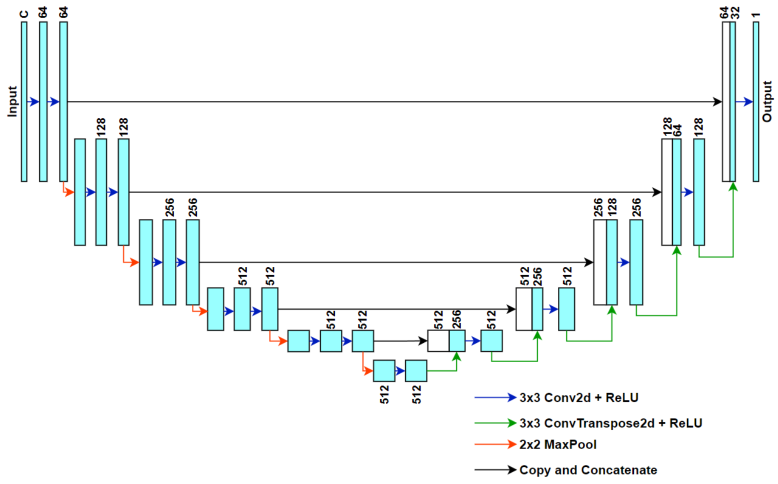


Figure A.17: U-Net architecture with input channels $= C$. C is the number of input channels, which, in our case, equals the number of ensemble members plus all climate data.

Our U-Net differs from the original U-Net by modifying the first 2D convolutional layer after input. Since our input channels can be different when we choose a different subset of features or different ensemble (NCEP-CFSv2 or NASA-GMAO), this 2D convolutional layer is used to transform our input with C channels into a latent representation with 64 channels. The number of channels C depends on which ensemble we are using and what task we are performing. For example, for precipitation tasks using the NCEP-CFSv2 ensemble, the input channels include 24 ensemble members, 5 lagged observations, 4 other observational variables, 8 principal components of SSTs, and 24 positional encodings, resulting in 65 channels in total. For temperature tasks using the NASA-GMAO ensemble, there are only 11 ensemble members, and we don't include SSTs information, hence there are only 44 channels in total. The other following layers use the same configurations with the standard U-Net Ronneberger et al. [2015].

We also perform careful hyperparameter tuning for the U-NET. In particular, we run a 10-fold cross-validation on our training set, and use grid search for tuning learning rate, batch

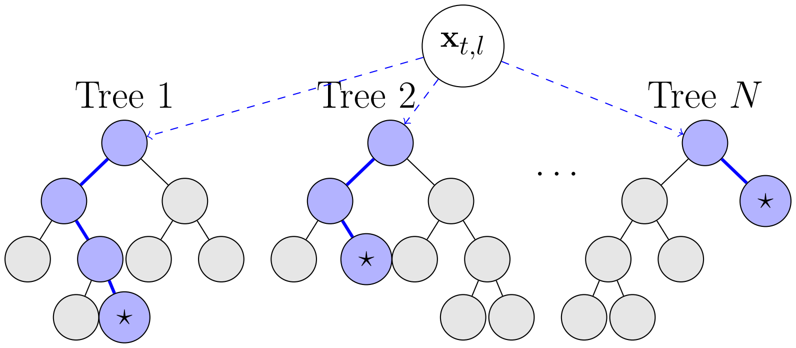


Figure A.18: Illustration of a random forest, which serves as a visual aid for our discussion. Quantile regression forests outputs the empirical α quantiles of the collection of all responses in all the leaves associated with $\mathbf{x}_{t,l}$ (marked with a star for each tree).

size, number of epochs, and weight decay. Since we use different loss functions for different forecast tasks and different numbers of input channels for NCEP-CFSv2 and GMAO-GMAO ensemble, we run hyperparameter tuning with the same cross-validation scheme separately for these tasks. For instance, for precipitation regression, we choose from 100, 120, 150, 170, 200, and 250 epochs; batch size may be equal to 8, 16, 32; learning rate values are chosen from 0.0001, 0.001, 0.01; weight decay can be 0, 0.001, 0.0001. In case of NCEP-CFSv2 precipitation regression, the optimal parameters are 170 epochs, batch size 16, learning rate 0.0001, and weight decay 0.0001. For temperature regression using the same data, the best parameters are 100 epochs, batch size 16, learning rate 0.001, and weight decay 0.001. For tercile classification of precipitation, the best parameters are 80 epochs (we chose from 60, 70, 80, 90, and 100 epochs during classification), batch size 8, learning rate 0.001, and weight decay 0.0001.

A.4.2 Random Forest Quantile Regressor details

We show a figure representation of the RFQR in Figure A.18. The RFQR is essentially trained as a regular random forest, but it makes a quantile estimate by taking the sample quantile of the responses in all leaves associated with a new input.

A.4.3 Stacking model details

The stacking model is a simple one-layer neural network with 100 hidden neurons and a sigmoid activation function for regression and softmax for classification. We use an implementation from Scikit-learn library [Pedregosa et al., 2011]. We choose 100 neurons based on the stacking model performance on the validation data (we also try 50, 75, 100 and 120 neurons). The stacking model demonstrates stable performance in general, but with 100 neurons it usually achieves the best results. We use the “lbfgs" optimizer from quasi-Newton methods for the regression tasks, and the SGD optimizer for classification tasks.

A.5 Preprocessing Details

Random forest and U-Net require different input formats. For U-Net, all input variables have natural image representation except SSTs and information about location. For example, ensemble predictions can be represented as a tensor of shape (K, W, H), where K corresponds to the number of ensemble members (or number of channels of an image), and W and H are width and height of the corresponding image. In our case, $W = 64$ and $H = 128$. For the U-Net model, we handle the missing land variables over the sea regions by the nearest neighbor interpolation of available values.

Sea surface temperatures There are more than 100,000 SSTs locations available. We extract the top eight principal components. Principal component analysis fits on the train part and then is applied to the rest of the data. In the case of U-Net, we deal with PCs of SSTs by adding additional input channels that are constant across space, with each channel corresponding to one of PCs. Random forest can use PCs from SSTs directly with no special preprocessing.

Normalization We apply channel-wise min-max normalization to the input features at each location based on the training part of the dataset in the case of U-Net. As for normalization of the true values, min-max normalization is applied for precipitation, and standardization is applied for temperature. This choice affects the final layer of the U-Net model, too: for the precipitation regression task, a sigmoid activation is used, and no activation function is applied for temperature regression. For the stacking model, we apply min-max normalization to both input and target values.

APPENDIX B
ADDITIONAL TOPICS IN TRAINING NEURAL OPERATORS TO PRESERVE INVARIANT MEASURES OF CHAOTIC ATTRACTORS

B.1 Additional Discussion

B.1.1 Contrastive feature learning vs. physics-informed optimal transport

Our two proposed approaches both aim to train neural operators to preserve the invariant measures of chaotic attractors. Ultimately, both methods have strong performance according to a variety of metrics, but the contrastive learning approach requires no prior physical knowledge and is often faster. Both approaches have significant advantages over the standard approach of minimizing RMSE, which fails to preserve important statistical characteristics of the system.

The primary conceptual difference between our approaches is that the optimal transport approach relies on prior domain knowledge, while the contrastive learning approach learns from the data alone. Specifically, the physics-informed optimal transport approach requires a choice of summary statistics, so it is much more dependent on our prior knowledge about the system and its chaotic attractor. Contrastive feature learning does not require a choice of statistics and instead learns useful invariant statistics on its own—although it may still be useful to have known relevant statistics to use as evaluation metrics for hyperparameter tuning.

The neural operators trained using the optimal transport loss perform the best on the L_1 histogram distance of the summary statistics $\mathbf{S}$ (Table 3.1 and 3.2), which is unsurprising given that the optimal transport loss specifically matches the distribution of $\mathbf{S}$. If we look at a different evaluation metric, e.g. the energy spectrum error or leading LE, the contrastive learning loss, in some cases, performs better even without prior knowledge. This suggests that contrastive feature learning may be capturing a wider range or different set of invariant statistics. We also see that the quality of the emulator degrades if we choose less informative statistics (Appendix B.2.3).

B.1.2 Interactions and trade-offs between short-term prediction and long-term statistics

Comparing our results in Tables 3.1 and 3.2, we find that emulators trained using our approaches perform significantly better on the long-term evaluation metrics (e.g. L_1 histogram distance and energy spectrum error) by trading off a bit of performance in terms of short-term RMSE (Tables B.9 and B.10). This suggests that training an emulator to model chaotic dynamics using purely RMSE can result in overfitting and generally a poor model of long-term dynamics. However, we still believe that RMSE is a useful part of the loss, even for long-term evaluation metrics, since it is the only term that is directly enforcing the short-term dynamics. In some cases, we find that the invariant statistics $\hat{S}_{\mathcal{A}}$ of trajectories generated by our trained neural operators are closer to the true statistics $S_{\mathcal{A}}$ than the statistics directly computed using the noisy training data $\tilde{S}_{\mathcal{A}}$, i.e., $|\hat{S}_{\mathcal{A}} - S_{\mathcal{A}}| < |\tilde{S}_{\mathcal{A}} - S_{\mathcal{A}}|$, despite being trained on the noisy data (Table B.1). We attribute this result to the RMSE component of our losses, which provides additional regularization by enforcing the short-term dynamics, combined with the inductive biases of the neural operator architectures, which are designed for modeling PDEs.

B.2 Additional Experiments and Evaluation Metrics

We have performed several additional experiments that act as points of comparison, help us better understand the behavior of our methods under a variety of conditions, and provide

Table B.1: Histogram error of neural operator predictions vs. noisy training data on Lorenz-96. Averaging over 200 testing instances with varying $\phi^{(n)}$, we compared the L_1 histogram error of the predicted dynamics from the neural operator $|\hat{S}_{\mathcal{A}} - S_{\mathcal{A}}|$ with the histogram error of the raw noisy training data $|\tilde{S}_{\mathcal{A}} - S_{\mathcal{A}}|$. This shows that, even though the neural operator is trained on the noisy data, the statistics of the predicted dynamics are often better than those computed directly from the noisy data. Here, $S_{\mathcal{A}}$ is computed from the noiseless ground truth data.

r	Training	Histogram Error $\lvert\hat{S}_{\mathcal{A}} - S_{\mathcal{A}}\rvert \downarrow$	Histogram Error $\lvert\tilde{S}_{\mathcal{A}} - S_{\mathcal{A}}\rvert \downarrow$
0.1	ℓ_{RMSE}	0.056 (0.051, 0.062)	0.029 (0.023, 0.036)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.029** (0.027, 0.032)	0.029 (0.023, 0.036)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.033 (0.029, 0.037)	0.029 (0.023, 0.036)
0.2	ℓ_{RMSE}	0.130 (0.118, 0.142)	0.101 (0.086, 0.128)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.039** (0.035, 0.042)	0.101 (0.086, 0.128)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.073 (0.066, 0.080)	0.101 (0.086, 0.128)
0.3	ℓ_{RMSE}	0.215 (0.204, 0.234)	0.213 (0.190, 0.255)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.057** (0.052, 0.064)	0.213 (0.190, 0.255)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.132 (0.111, 0.151)	0.213 (0.190, 0.255)

useful insights for future applications of our approaches.

B.2.1 Denoising with Gaussian blurring

Gaussian blurring, often used as a denoising technique for images, employs a Gaussian distribution to establish a convolution matrix that's applied to the original image. The fundamental idea involves substituting the noisy pixel with a weighted average of surrounding pixel values. A key hyperparameter in Gaussian blurring is the standard deviation of the Gaussian distribution. When the standard deviation approaches zero, it fundamentally indicates the absence of any blur. Under such circumstances, the Gaussian function collapses to a single point, leading to the elimination of the blur effect. In Table B.2, we present the results from applying Gaussian blurring to noisy data during training based solely on RMSE. Despite the effectiveness of the widely adopted denoising approach, our findings indicate that Gaussian blurring may not be ideally suited for our purpose of emulating dynamics. This is primarily because significant invariant statistics might be strongly correlated with certain high-frequency signals that could be affected by the blurring preprocessing.

Table B.2: Emulator performance with Gaussian blurring on Kuramoto–Sivashinsky data with noise scale $r = 0.3$. Averaging over 200 testing instances with varying $\phi^{(n)}$, we show the performance of (1) the application of Gaussian blurring as a preliminary denoising effort with a small standard deviation ($\sigma_b = 0.1$); (2) the application of Gaussian blurring with a larger standard deviation ($\sigma_b = 0.5$); and (3) training purely on RMSE without any blurring preprocessing. The results suggest that the application of Gaussian blurring might further degrade the results, as the high-frequency signals associated with invariant statistics can be lost.

Training	Histogram Error $\downarrow$	Energy Spec. Error$\downarrow$	Leading LE Error $\downarrow$
ℓ_{RMSE} ($\sigma_b = 0.1$)	**0.390** (0.326, 0.556)	**0.290** (0.226, 0.402)	**0.098** (0.069, 0.127)
ℓ_{RMSE} ($\sigma_b = 0.5$)	1.011 (0.788, 1.264)	0.493 (0.379, 0.623)	**0.098** (0.041, 0.427)
ℓ_{RMSE}	**0.390** (0.325, 0.556)	**0.290** (0.225, 0.402)	0.101 (0.069, 0.122)

B.2.2 Sobolev norm baseline

We recognize that there are alternative methods that strive to capture high-frequency signals by modifying training objectives. For instance, the Sobolev norm, which combines data and its derivatives, has been found to be quite effective in capturing high-frequency signals [Zhu et al., 2021a, Li et al., 2022b]. However, its effectiveness can be significantly curtailed in a noisy environment, especially when noise is introduced to a high-frequency domain, as minimizing the Sobolev norm then fails to accurately capture relevant statistics, as shown in Tables B.3 and B.4.

B.2.3 Optimal transport: reduced set of summary statistics

For our optimal transport approach, we test a reduced set of summary statistics, which shows how the quality of the summary statistic affects the performance of the method (Table B.5). With an informative summary statistic, we find even a reduced set can still be helpful but, for a non-informative statistic, the optimal transport method fails as expected.

Table B.3: Emulator performance (including Sobolev norm loss) on Lorenz-96 data with noise scale $r = 0.3$. The median (25th, 75th percentile) of the evaluation metrics are computed on 200 Lorenz-96 test instances (each with 1500 time steps) for the neural operator trained with (1) only RMSE loss ℓ_{RMSE}; (2) Sobolev norm loss with dissipative regularization $\ell_{\text{Sobolev}} + \ell_{\text{dissaptive}}$; (3) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$; and (4) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$.

Training	Histogram Error ↓	Energy Spec. Error ↓	Leading LE Error ↓	FD Error ↓
ℓ_{RMSE}	0.215 (0.204, 0.234)	0.291 (0.280, 0.305)	0.440 (0.425, 0.463)	3.580 (2.333, 4.866)
$\ell_{\text{Sobolev}} + \ell_{\text{dissaptive}}$	0.246 (0.235, 0.255)	0.325 (0.341, 0.307)	0.487 (0.456, 0.545)	4.602 (3.329, 6.327)
$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.057** (0.052, 0.064)	**0.123** (0.116, 0.135)	0.084 (0.062, 0.134)	3.453 (2.457, 4.782)
$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.132 (0.111, 0.151)	0.241 (0.208, 0.285)	**0.064** (0.045, 0.091)	**1.894** (0.942, 3.108)

Table B.4: Emulator performance (including Sobolev norm loss) on Kuramoto–Sivashinsky data with noise scale $r = 0.3$. The median (25th, 75th percentile) of the evaluation metrics are computed on 200 Kuramoto–Sivashinsky test instances (each with 1000 time steps) for the neural operator trained with (1) only RMSE loss ℓ_{RMSE}; (2) Sobolev norm loss with dissipative regularization $\ell_{\text{Sobolev}} + \ell_{\text{dissaptive}}$; (3) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$; and (4) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$.

Training	Histogram Error ↓	Energy Spec. Error ↓	Leading LE Error ↓
ℓ_{RMSE}	0.390 (0.325, 0.556)	0.290 (0.225, 0.402)	0.101 (0.069, 0.122)
$\ell_{\text{Sobolev}} + \ell_{\text{dissipative}}$	0.427 (0.289, 0.616)	0.237 (0.204, 0.315)	**0.023** (0.012, 0.047)
$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.172** (0.146, 0.197)	0.211 (0.188, 0.250)	0.094 (0.041, 0.127)
$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.193 (0.148, 0.247)	**0.176** (0.130, 0.245)	0.108 (0.068, 0.132)

Table B.5: Emulator performance for different choices of summary statistics on Lorenz-96 data with noise scale $r = 0.3$. Each neural operator was trained using the optimal transport and RMSE loss using (1) full statistics $\mathbf{S}(\mathbf{u}) := \{\frac{du_i}{dt}, (u_{i+1} - u_{i-2})u_{i-1}, u_i\}$; (2) partial statistics $\mathbf{S}_1(\mathbf{u}) := \{(u_{i+1} - u_{i-2})u_{i-1}\}$; or (3) minimum statistics $\mathbf{S}_2(\mathbf{u}) := \{\bar{\mathbf{u}}\}$, where $\bar{\mathbf{u}}$ is the spatial average.

Training statistics	Histogram Error ↓	Energy Spec. Error ↓	Leading LE Error ↓	FD Error ↓
$\mathbf{S}$ (full)	**0.057** (0.052, 0.064)	**0.123** (0.116, 0.135)	0.084 (0.062, 0.134)	3.453 (2.457, 4.782)
$\mathbf{S}_1$ (partial)	0.090 (0.084, 0.098)	0.198 (0.189, 0.208)	0.263 (0.217, 0.323)	3.992 (2.543, 5.440)
$\mathbf{S}_2$ (minimum)	0.221 (0.210, 0.234)	0.221 (0.210, 0.230)	0.276 (0.258, 0.291)	**3.204** (2.037, 4.679)

B.2.4 Contrastive learning: reduced environment diversity

For our contrastive learning approach, we test a multi-environment setting with reduced data diversity and find that the contrastive method still performs well under the reduced conditions (Table B.6), which demonstrates robustness.

Table B.6: Emulator performance with reduced environment diversity (i.e. narrower parameter range) on Lorenz-96 data with noise level $r = 0.3$. Averaging over 200 testing instances, we show the performance of training the neural operator with (1) only RMSE loss ℓ_{RMSE}; (2) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$; and (3) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$. We shrink the parameter range for generating the dataset from $[10, 18]$ to $[16, 18]$.

Training	Histogram Error ↓	Energy Spec. Error ↓	Leading LE Error ↓	FD Error ↓
ℓ_{RMSE}	0.255 (0.248, 0.263)	0.307 (0.302, 0.315)	0.459 (0.743, 2.746)	3.879 (2.456, 5.076)
$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.055** (0.050, 0.061)	**0.124** (0.116, 0.131)	0.080 (0.045, 0.109)	4.015 (2.401, 5.225)
$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.130 (0.111, 0.152)	0.193 (0.183, 0.200)	**0.031** (0.014, 0.053)	**1.747** (0.792, 2.939)

B.2.5 Maximum mean discrepancy (MMD) vs. optimal transport loss

We also implement a variant of our optimal transport approach that uses maximum mean discrepancy (MMD) as a distributional distance rather than the Sinkhorn divergence. Using the same set of summary statistics, we find that MMD does not perform as well as our optimal transport loss for training emulators (Table B.7).

Table B.7: Emulator performance (including MMD loss) on Kuramoto–Sivashinsky data with noise scale $r = 0.3$. The median (25th, 75th percentile) of the evaluation metrics are computed on 200 Kuramoto–Sivashinsky test instances (each with 1000 time steps) for the neural operator trained with (1) only RMSE loss ℓ_{RMSE}; (2) maximum mean discrepency (MMD) and RMSE loss $\ell_{\text{MMD}} + \ell_{\text{RMSE}}$; (3) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$; and (4) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$.

Training	Histogram Error ↓	Energy Spec. Error ↓	Leading LE Error ↓
ℓ_{RMSE}	0.390 (0.325, 0.556)	0.290 (0.225, 0.402)	0.101 (0.069, 0.122)
$\ell_{\text{MMD}} + \ell_{\text{RMSE}}$	0.245 (0.218, 0.334)	0.216 (0.186, 0.272)	0.101 (0.058, 0.125)
$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	**0.172** (0.146, 0.197)	0.211 (0.188, 0.250)	**0.094** (0.041, 0.127)
$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.193 (0.148, 0.247)	**0.176** (0.130, 0.245)	0.108 (0.068, 0.132)

B.2.6 Additional Lyapunov spectrum evaluation metrics

In the table B.8, we evaluated the results of Lorenz 96 on Lyapunov spectrum error rates and the total number of positive Lyapunov exponents error rates. For the Lyapunov spectrum error, we report the sum of relative absolute errors across the full spectrum: $\sum_i^d |\hat{\lambda}_i - \lambda_i|/\lambda_i$, where λ_i is the i-th Lyapunov exponent and d is the dimension of the dynamical state. As suggested by [Wang et al., 2018], we also compare the number of positive Lyapunov exponents (LEs) as an additional statistic to measure the complexity of the chaotic dynamics. We compute the absolute error in the number of positive LEs $\sum_i^d |\mathbf{1}(\hat{\lambda_i} > 0 - \mathbf{1}(\lambda_i > 0)|$.

Table B.8: Emulator performance on Lyapunov spectrum metrics for Lorenz-96 data. The median (25th, 75th percentile) of the Lyapunov spectrum metrics are computed on 200 Lorenz-96 test instances (each with 1500 time steps) for the neural operator trained with (1) only RMSE loss ℓ_{RMSE}; (2) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$; and (3) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$. In the presence of high noise, OT and CL give lower relative errors on the leading Lyapunov exponent (LE). When evaluating the full Lyapunov spectrum, OT and CL show significant advantages than the baseline. In addition, the lower absolute errors of the total number of the positive Lyapunov exponents (LEs) suggest that OT and CL are able to match the complexity of the true chaotic dynamics.

r	Training	Leading LE Error ↓	Lyapunov Spectrum Error ↓	Total number of positive LEs Error ↓
0.1	ℓ_{RMSE}	**0.013** (0.006, 0.021)	0.265 (0.110, 0.309)	0.500 (0.000, 1.000)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	0.050 (0.040, 0.059)	0.248 (0.168, 0.285)	**0.000** (0.000, 1.000)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.065 (0.058, 0.073)	**0.227** (0.164, 0.289)	**0.000** (0.000, 1.000)
0.2	ℓ_{RMSE}	0.170 (0.156, 0.191)	0.612 (0.522, 0.727)	4.000 (4.000, 5.000)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	0.016 (0.006, 0.030)	0.513 (0.122, 0.590)	**3.000** (2.000, 3.000)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	**0.012** (0.006, 0.018)	**0.459** (0.138, 0.568)	**3.000** (2.000, 3.000)
0.3	ℓ_{RMSE}	0.440 (0.425, 0.463)	0.760 (0.702, 0.939)	7.000 (7.000, 8.000)
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	0.084 (0.062, 0.134)	**0.661** (0.572, 0.746)	**5.000** (4.000, 6.000)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	**0.064** (0.045, 0.091)	0.654 (0.558, 0.780)	6.000 (5.000, 6.000)

B.2.7 1-step RMSE evaluation results

Evaluating on 1-step RMSE only shows short-term prediction performance and is not an informative evaluation metric for long-term behavior. Here, we report the 1-step RMSE (Tables B.9 and B.10) to show that training using our approaches retains similar 1-step RMSE results while significantly improving on long-term statistical metrics (Tables 3.1 and 3.2). See Appendix B.1.2 for additional discussion.

Table B.9: 1-step RMSE performance on Lorenz-96 data. The median (25th, 75th percentile) of the Lyapunov spectrum metrics are computed on 200 Lorenz-96 test instances (each with 1500 time steps) for the neural operator trained with (1) only RMSE loss ℓ_{RMSE}; (2) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$; and (3) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$. We see comparable performance on short-term 1-step RMSE.

r	Training	RMSE $\downarrow$
	ℓ_{RMSE}	0.107 (0.105, 0.109)
0.1	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	0.108 (0.105, 0.110)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.109 (0.108, 0.113)
	ℓ_{RMSE}	0.202 (0.197, 0.207)
0.2	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	0.207 (0.202, 0.212)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.214 (0.203, 0.218)
	ℓ_{RMSE}	0.288 (0.282, 0.296)
0.3	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	0.301 (0.293, 0.307)
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.312 (0.302,0.316)

Table B.10: 1-step RMSE performance on Kuramoto–Sivashinsky data with noise scale $r = 0.3$. The median (25th, 75th percentile) of the evaluation metrics are computed on 200 Kuramoto–Sivashinsky test instances (each with 1000 time steps) for the neural operator trained with (1) only RMSE loss ℓ_{RMSE}; (2) optimal transport (OT) and RMSE loss $\ell_{\text{OT}} + \ell_{\text{RMSE}}$; and (3) contrastive learning (CL) and RMSE loss $\ell_{\text{CL}} + \ell_{\text{RMSE}}$. We again see comparable performance on short-term 1-step RMSE.

Training	RMSE $\downarrow$
ℓ_{RMSE}	0.373 (0.336, 0.421)
$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	0.381 (0.344, 0.430)
$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	0.402 (0.364, 0.452)

B.3 Implementation Details

B.3.1 Evaluation metrics

To evaluate the long-term statistical behavior of our trained neural operators, we run the neural operator in a recurrent fashion for 1500 (Lorenz-96) or 1000 (Kuramoto–Sivashinsky) time steps and then compute our long-term evaluation metrics on this autonomously generated time-series.

Histogram error. For distributional distance with a pre-specified statistics $\mathbf{S}$, we first compute the histogram of the invariant statistics $\mathbf{H}(\mathbf{S}) = \{(\mathbf{S}_1, c_1), (\mathbf{S}_2, c_2), \ldots, (\mathbf{S}_B, c_B)\}$, where $\mathbf{H}$ represents the histogram, and the values of the bins are denoted by $\mathbf{S}_b$ with their corresponding frequencies c_b. We then define the L_1 histogram error as:

$$\mathrm{Err}(\hat{\mathbf{H}}, \mathbf{H}) := \sum_{b=1}^{B} \|c_b - \hat{c}_b\|_1. \tag{B.1}$$

Note that for a fair comparison across all our experiments, we use the rule of thumb—the square root rule to decide the number of bins.

Energy spectrum error. We compute the relative mean absolute error of the energy spectrum—the squared norm of the spatial FFT $\mathcal{F}[\mathbf{u}_t]$—averaged over time:

$$\frac{1}{T} \sum_{\mathbf{u}_t, \hat{\mathbf{u}}_t \in \mathbf{U}_{1:T}, \hat{\mathbf{U}}_{1:T}} \frac{\||\mathcal{F}[\mathbf{u}_t]|^2 - |\mathcal{F}[\hat{\mathbf{u}}_t]|^2\|_1}{\||\mathcal{F}[\mathbf{u}_t]|^2\|_1}. \tag{B.2}$$

Leading LE error. The leading Lyapunov exponent (LE) is a dynamical invariant that measures how quickly the chaotic system becomes unpredictable. For the leading LE error, we report the relative absolute error $|\hat{\lambda} - \lambda|/|\lambda|$ between the model and the ground truth averaged over the test set. We adapted the Julia DynamicalSystem.jl package to calculate the leading LE.

FD error. The fractal dimension (FD) is a characterization of the dimension of the attractor. We report the absolute error $|\hat{D} - D|$ between the estimated FD of the model and the ground truth averaged over the test set. We use the Julia DynamicalSystem.jl package for calculating the fractal dimension.

RMSE. We use 1-step relative RMSE $\|\mathbf{u}_{t+\Delta t} - \hat{g}_\theta(\mathbf{u}_t, \phi)\|_2 / \|\mathbf{u}_{t+\Delta t}\|_2$ to measure short-term prediction accuracy.

B.3.2 Time complexity

The optimal transport approach relies on the Sinkhorn algorithm which scales as $\mathcal{O}(n^2 \log n)$ for comparing two distributions of n points each (Theorem 2 in Dvurechensky et al. [2018]).

In our experiments, we use $n = 6000$ to $n = 25600$ points with no issues, so this approach scales relatively well. The contrastive learning approach requires pretraining but is even faster during emulator training since it uses a fixed, pre-trained embedding network.

Table B.11: Empirical training time. Training time (minutes) with 4 GPUs for 60-dimensional Lorenz-96 and 256-dimensional Kuramoto–Sivashinsky.

	Training	Encoder	Operator	Total
Lorenz-96	ℓ_{RMSE}	–	20	20
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	–	51	51
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	22	27	49
Kuramoto–Sivashinsky	ℓ_{RMSE}	–	55	55
	$\ell_{\text{OT}} + \ell_{\text{RMSE}}$	–	262	262
	$\ell_{\text{CL}} + \ell_{\text{RMSE}}$	26	56	82

B.3.3 Training of the encoder

Evaluation rule. A standard procedure for assessing the performance of an encoder trained with Noise Contrastive Estimation (InfoNCE) loss in an unsupervised manner [Chen et al., 2020, He et al., 2020a, Wu et al., 2018], is employing the top-1 accuracy metric. This measures how effectively similar items are positioned closer to each other compared to dissimilar ones in the embedded space. While this evaluation measure is commonly employed in downstream tasks that focus on classification, it is suitably in line with our goals. We aim to learn an encoder that can differentiate whether sequences of trajectories are sampled from different attractors, each characterized by distinct invariant statistics. Therefore, the use of Top-1 accuracy to evaluate if two sequences originate from the same trajectory serves our purpose effectively. And we utilize this metric during our training evaluations to assess the performance of our encoder.

The formal definition of Top-1 accuracy requires us to initially define the softmax function σ, which is used to estimate the likelihood that two samples from different time windows originate from the same trajectory,

$$\sigma\left(f_\psi(\mathbf{U}^{(j)}_{J:J+K}); f_\psi(\mathbf{U}^{(n)}_{I:I+K})\right) = \frac{\exp\left(\langle f_\psi(\mathbf{U}^{(n)}_{I:I+K}), f_\psi(\mathbf{U}^{(j)}_{J:J+K})\rangle\right)}{\sum_{m=1}^{N}\left[\exp\left(\langle f_\psi(\mathbf{U}^{(n)}_{I:I+K}), f_\psi(\mathbf{U}^{(m)}_{H:H+K})\rangle\right)\right]}. \tag{B.3}$$

Using the softmax function to transform the encoder's output into a probability distribution, we then define Top-1 accuracy as:

$$\mathrm{Acc}_1 = \frac{1}{N}\sum_{n=1}^{N}\mathbf{I}\left\{n = \arg\max_j \sigma\left(f_\psi(\mathbf{U}^{(j)}_{J:J+K}); f_\psi(\mathbf{U}^{(n)}_{I:I+K})\right)\right\}, \tag{B.4}$$

where $\mathbf{1}(\cdot)$ is the indicator function representing whether two most close samples in the feature space comes from the same trajectory or not.

Training hyperparameters. We use the ResNet-34 as the backbone of the encoder, throughout all experiments, we train the encoder using the AdamW optimization algorithm Loshchilov and Hutter [2019], with a weight decay of 10^{-5}, and set the training duration to 2000 epochs.

For the temperature value τ balancing the weights of difficult and easy-to-distinguish samples in contrastive learning, we use the same warm-up strategy as in [Jiang and Willett, 2022]. Initially, we start with a relatively low τ value (0.3 in our experiments) for the first 1000 epochs. This ensures that samples that are difficult to distinguish get large gradients. Subsequently, we incrementally increase τ up to a specified value (0.7 in our case), promoting the grouping of similar samples within the embedded space. From our empirical observations, we have found that this approach leads to an improvement in Top-1 accuracy in our experiments.

The length of the sequence directly influences the Top-1 accuracy. Considering that the sample length L of the training data $\{\mathbf{U}^{(n)}_{0:L}\}_{n=1}^{N}$ is finite, excessively increasing the crop length K can have both advantages and disadvantages. On one hand, it enables the encoder to encapsulate more information; on the other hand, it could lead to the failure of the encoder's training. This is likely to occur if two samples, i.e., $f_\psi(\mathbf{U}^{(n)}_{I:I+K})$ and $f_\psi(\mathbf{U}^{(n)}_{J:J+K})$, from the same trajectory overlap excessively, inhibiting the encoder from learning a meaningful feature space. With this consideration, we've empirically chosen the length K of subsequences for the encoder to handle to be approximately 5% of the total length T.

B.3.4 Lorenz-96

Data generation. To better align with realistic scenarios, we generate our training data with random initial conditions drawn from a normal distribution. For the purpose of multi-environment learning, we generate 2000 trajectories for training. Each of these trajectories has a value of $\phi^{(n)}$ randomly sampled from a uniform distribution ranging from 10.0 to 18.0. We discretize these training trajectories into time steps of $dt = 0.1$ over a total time of $t = 205s$, yielding 2050 discretized time steps. Moreover, in line with the setup used in [Li et al., 2021, Lu et al., 2021], we discard the initial 50 steps, which represent the states during an initial transient period.

Training hyperparameters. We determine the roll-out step during training (i.e., h and h_{RMSE}) via the grid search from the set of values $\{1, 2, 3, 4, 5\}$. Though it is shown in some cases that a larger roll-out number help improve the results [Gupta and Brandstetter, 2022], we have not observed that in our training. We hypothesized that this discrepancy may be due to our dataset being more chaotic compared to typical cases. Consequently, to ensure a fair comparison and optimal outcomes across all experiments, we have decided to set h and h_{RMSE} to 1.

In the case of the optimal transport loss[1], described in Section 3.3.1, the blur parameter γ governs the equilibrium between faster convergence speed and the precision of the Wasserstein distance approximation. We determined the value of γ through a grid search, examining values ranging from $\{0.01, 0.02, \ldots, 0.20\}$. Similarly, we decided the weights of the

1. https://www.kernel-operations.io/geomloss/

optimal transport loss α through a grid search, exploring values from $\{0.5, 1, 1.5, \ldots, 3.0\}$. For the experiments conducted using Lorenz-96, we selected $\alpha = 3$ and $\gamma = 0.02$.

In the context of the feature loss, the primary hyperparameter we need to consider is its weights, represented as λ. Given that we do not have knowledge of the invariant statistics $\mathbf{S}$ during the validation phase, we adjust λ according to a specific principle: our aim is to reduce the feature loss ℓ_{feature} (as defined in Eqn. 3.17) as long as it does not adversely affect the RMSE beyond a predetermined level (for instance, 10%) when compared with the baseline model, which is exclusively trained on RMSE. As illustrated in Figure B.1, we adjust the values of λ in a systematic grid search from $\{0.2, 0.4, 0.6, 0.8, 1.0, 1.2\}$. From our observations during the validation phase, we noted that the feature loss was at its lowest with an acceptable RMSE (which is lower than 110% compared to the baseline) when $\lambda = 0.8$. Therefore, we have reported our results with λ set at 0.8.

Results visualization. We present more results with varying noise levels in Figures B.2, B.3, B.4.

B.3.5 Kuramoto–Sivashinsky

Data generation. In order to ascertain that we are operating within a chaotic regime, we set the domain size $L = 50$ and the spatial discretization grids number $d = 256$. We select initial conditions randomly from a uniform range between $[-\pi, \pi]$. A fourth-order Runge-Kutta method was utilized to perform all simulations. We generate 2000 trajectories for training, with each $\phi^{(n)}$ being randomly chosen from a uniform distribution within the interval $[1.0, 2.6]$.

Training hyperparameters. In the case of the optimal transport loss, similar to the discussion in B.3.4, we search the blur value γ and the weights of loss α from a grid search. And in our experiment, we set $\alpha = 3$ and $\gamma = 0.05$. In the case of the feature loss, we again adopt the same rule for deciding the value of λ, where we compare the trend of feature loss

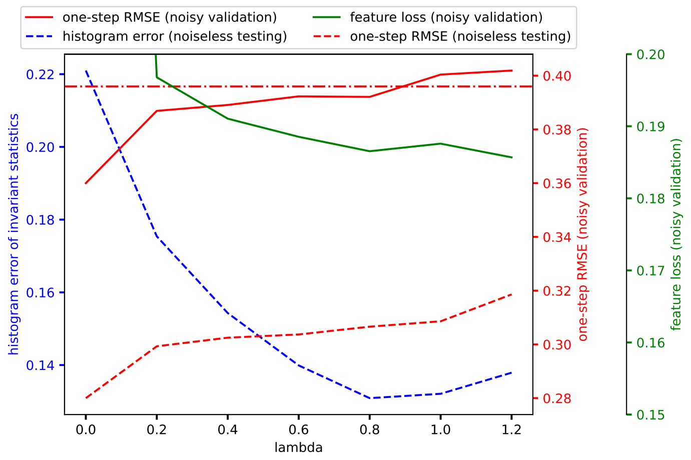
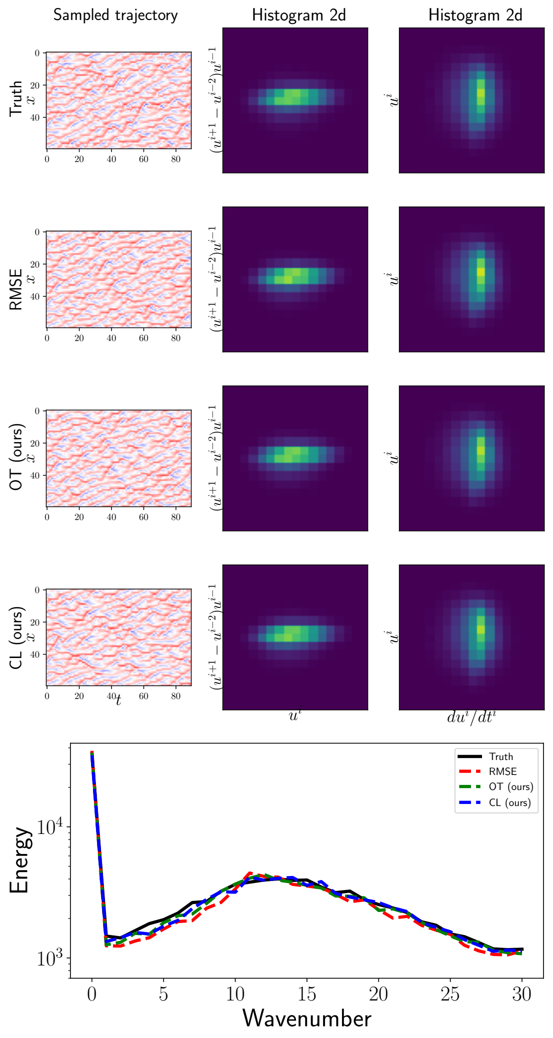
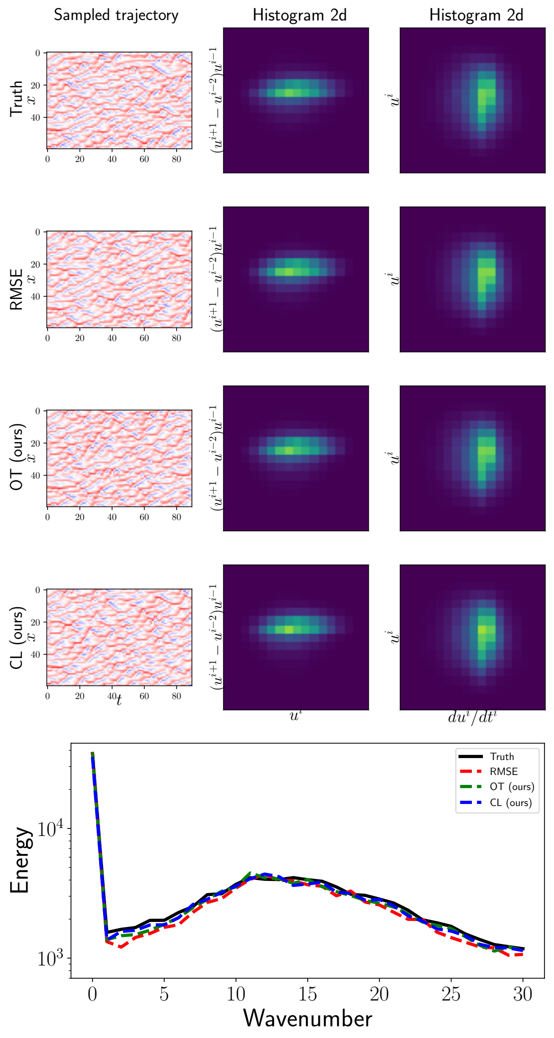


Figure B.1: The trend of feature loss with its weight λ when the scale of noise is $r = 0.3$. The solid lines in the figure represent the evaluation metrics during the validation phase, comparing the outputs of the neural operator to the noisy data. In contrast, the dashed lines represent the actual metrics we are interested in, comparing the outputs of the neural operator to the clean data and calculating the error of the invariant statistics. In addition, the horizontal solid dashed line correspond to the bar we set for the RMSE, i.e., 110% of the RMSE when $\lambda = 0$. We observe that, (1) with the increase of λ from 0, the feature loss decreases until λ reaches 1.0. (2) The RMSE generally increases with the increase of λ. (3) The unseen statistical error generally decreases with the increase of λ. We reported the results when $\lambda = 0.8$ as our final result, since the further increase of λ does not bring further benefit in decreasing the feature loss, and the result remains in an acceptable range in terms of RMSE.

with the relative change of RMSE of the baseline and choose to report the results with the lowest feature loss and acceptable RMSE increase (110% compared to the baseline when $\lambda = 0$).

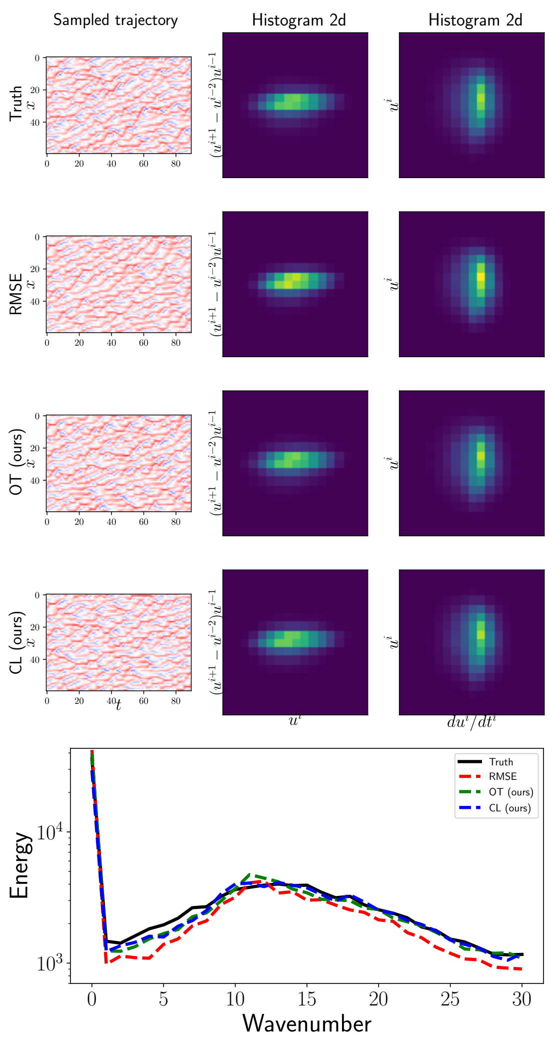


(a)

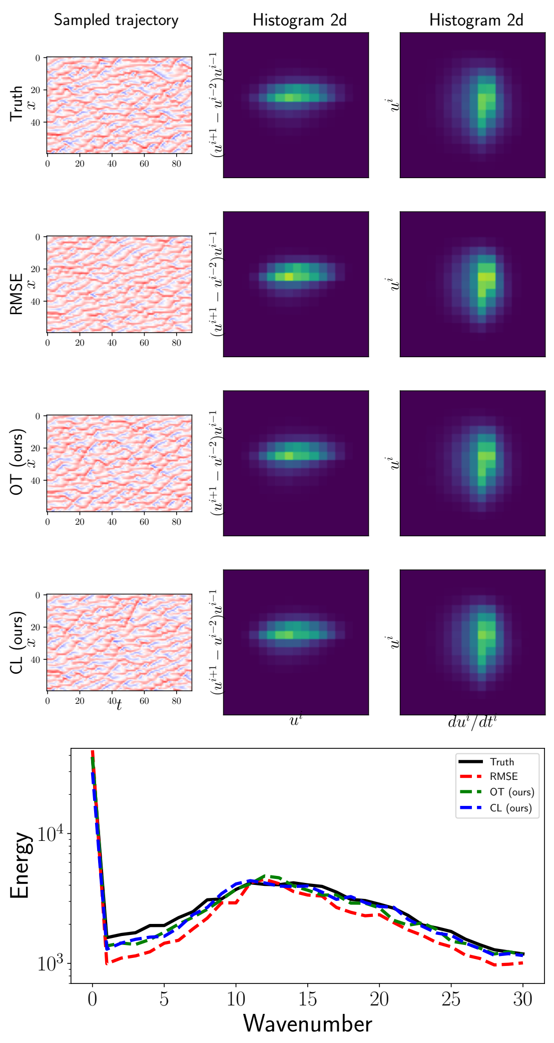


(b)

Figure B.2: Visualization of the predictions when the noise level $r = 0.1$. We evaluate our method by comparing them to the baseline that is trained solely using RMSE. For two different instances (a) and (b), we visualize the visual comparison of the predicted dynamics (left), two-dimensional histograms of relevant statistics (middle and right). We notice that, with the minimal noise, the predictions obtained from all methods look statistically consistent to the true dynamics.

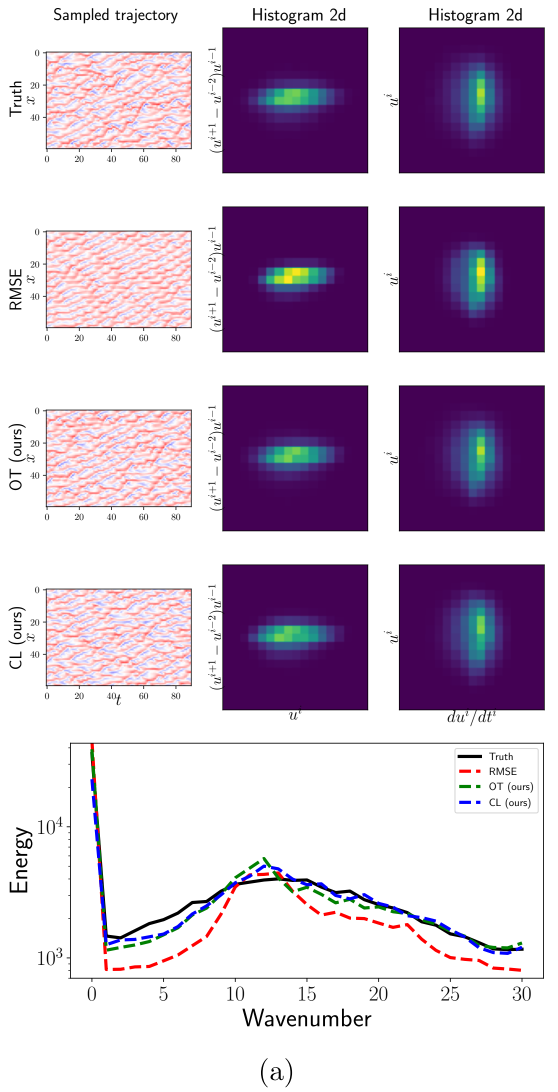


(a)



(b)

Figure B.3: Visualization of the predictions when the noise level $r = 0.2$. We evaluate our method by comparing them to the baseline that is trained solely using RMSE. For two different instances (a) and (b), we visualize the visual comparison of the predicted dynamics (left), two-dimensional histograms of relevant statistics (middle and right). We observe that as the noise level escalates, the degradation of performance in the results of RMSE is more rapid compared to our method, which employs optimal transport and feature loss during training.



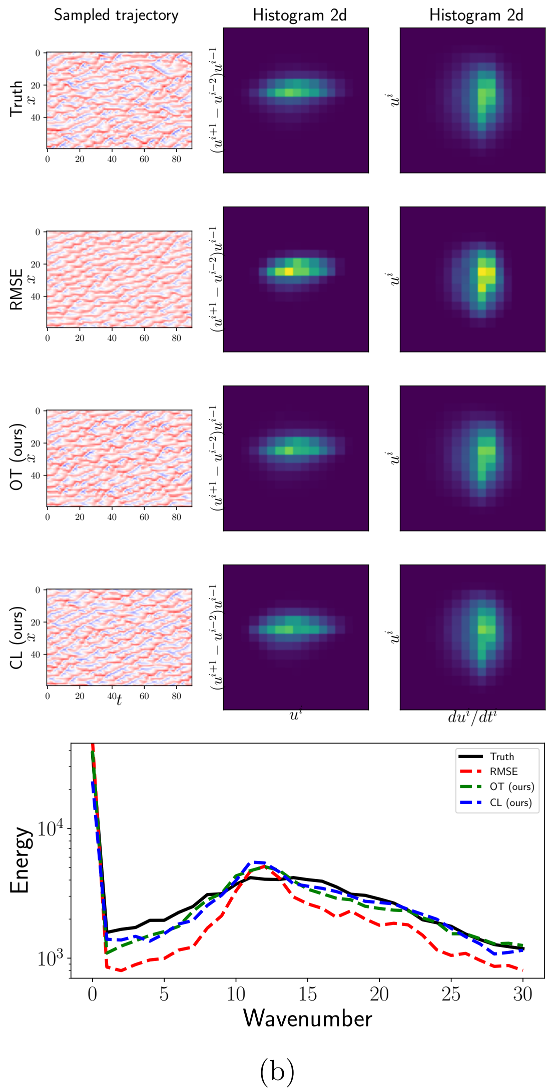


Figure B.4: Visualization of the predictions when the noise level $r = 0.3$. We evaluate our method by comparing them to the baseline that is trained solely using RMSE. For two different instances (a) and (b), we visualize the visual comparison of the predicted dynamics (left), two-dimensional histograms of relevant statistics (middle and right). We find that, under higher levels of noise, the RMSE results exhibit fewer negative values, as indicated by the blue stripes in the predicted dynamics. This is further confirmed by the energy spectrum, which clearly shows that the RMSE results are significantly deficient in capturing high-frequency signals.

APPENDIX C

ADDITIONAL TOPICS IN DSM

C.1 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 $\{x\}$.

C.2 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 (4.7a) 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, $\psi_0$ – 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 $X_{0j} \sim |\psi_0|^2$ 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** $\left\{\{X_{ij}\}_{j=1}^{B}\right\}_{i=0}^{N}$
```

C.3 Experiment Setup Details

C.3.1 Non-interacting system

In our experiments, we set $m = 1$, $\hbar = 10^{-2}$[1], $\sigma^2 = 10^{-1}$. 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

1. 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.

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 $u_{\theta_M}, v_{\theta_M}$
```

the observables are estimated from these samples; we run the model this way ten times.

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.

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.

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 4.3.4. 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 4.3.4), the number of hidden neurons is fixed to 200 for all dimensions. We use the Adam optimizer [Kingma and 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 4.3.4, we use an NVIDIA A100 GPU.

On optimization

We use Adam optimizer [Kingma and Ba, 2014] in our experiments. Since the operators in Equation (4.8) 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. [2022b] due to the linearity of the Schrödinger equation (4.1). It is possible that non-linearity in the loss from Equation (4.14) 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., 2022b].

The only noise source in our loss Equation (4.14) 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 (4.8) avoiding such contributions; for details, see Nelson [1966a, 2005]. This leads to $\mathcal{O}(1)$ variance of the gradient and, thus, allows us to achieve fast convergence with smaller batches.

C.3.2 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 [2]. 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.

Architecture and training details

Instead of a multi-layer perceptron, we follow the design choice of Jiang and 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 and Ba, 2014] with a learning rate 10^{-4} in our experiments. We use loss weights $w_u = 1, w_v = 1, w_0 = 1$.

2. https://github.com/quantum-visualizations/qmsolve

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 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.

C.4 Experimental Results

C.4.1 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 C.1 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 C.1 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.

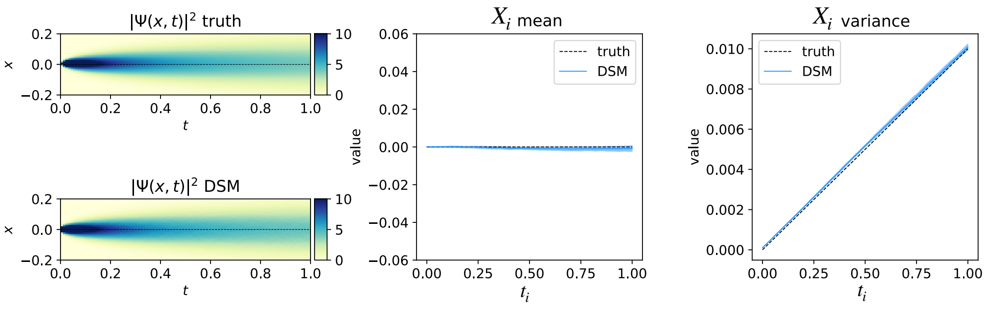


Figure C.1: Results for the singular initial condition problem. DSM corresponds to our method.

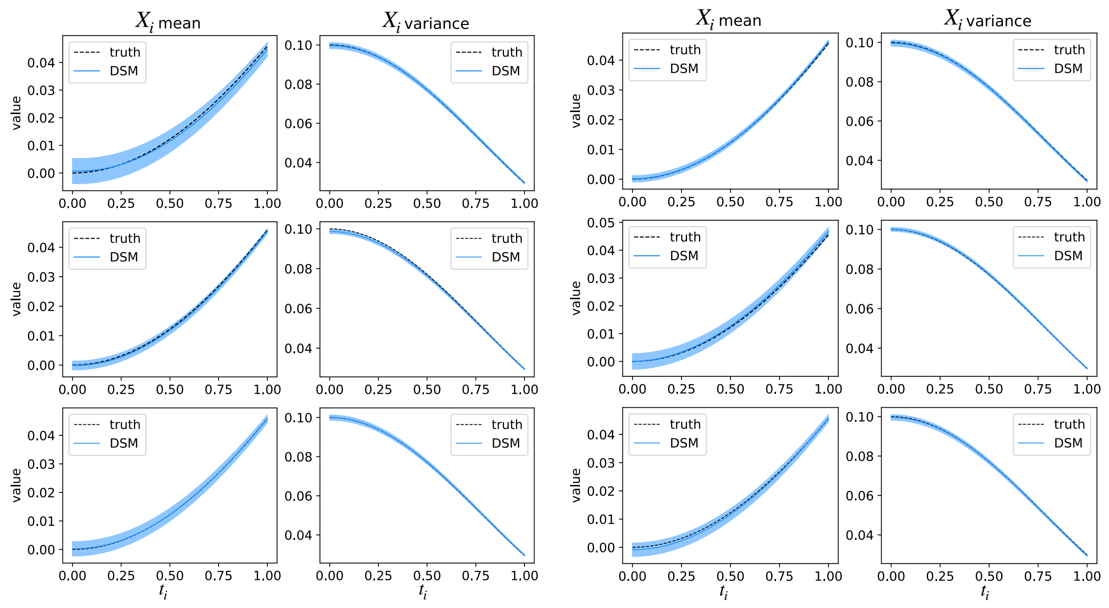
C.4.2 3D harmonic oscillator

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 4.2 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 C.2 demonstrates the obtained statistics with the proposed algorithm's two versions (Nelsonian and Gradient Divergence) for every dimension. Figure C.3 compares the density function for every dimension for these two versions. Table C.1 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 cubiccomplexity of the Nelsonian version.

C.4.3 Naive sampling

Figure C.4 shows the performance of the Gaussian sampling approach applied to the harmonic oscillator and the singular initial condition setting. Table C.2 compares results of all methods. Our approach converges to the ground truth while naive sampling does not.

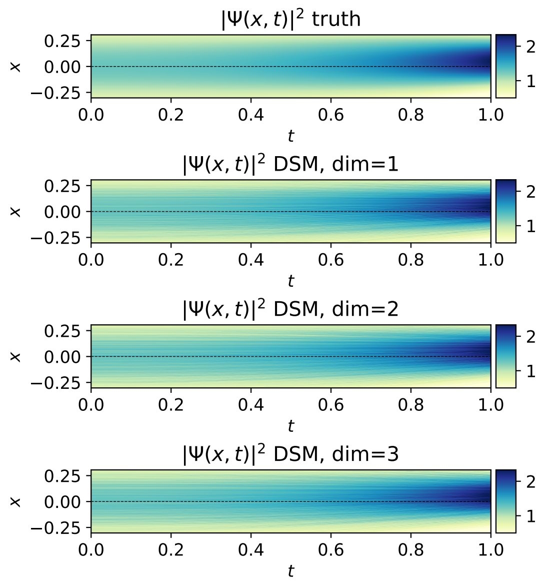


(a) The Nelsonian version. (b) The Gradient Divergence version.

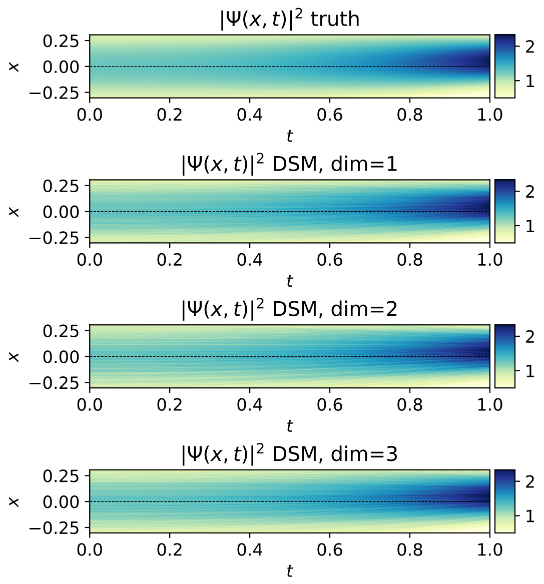
Figure C.2: The obtained statistics for 3d harmonic oscillator using two versions of the proposed approach.

Table C.1: 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	$\mathbf{0.073 \pm 0.072}$	0.100 ± 0.061
DSM (Gradient Divergence)	$\mathbf{0.038 \pm 0.023}$	$\mathbf{0.100 \pm 0.060}$	0.082 ± 0.060	$\mathbf{0.073 \pm 0.048}$
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)	$\mathbf{0.012 \pm 0.009}$	0.012 ± 0.009	0.011 ± 0.008	0.012 ± 0.009
DSM (Gradient Divergence)	$\mathbf{0.012 \pm 0.010}$	$\mathbf{0.009 \pm 0.005}$	$\mathbf{0.011 \pm 0.010}$	$\mathbf{0.011 \pm 0.008}$
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)	$\mathbf{4.346 \times 10^{-5}}$	$\mathbf{4.401 \times 10^{-5}}$	$\mathbf{4.700 \times 10^{-5}}$	$\mathbf{4.482 \times 10^{-5}}$
Model	$\mathcal{E}(u^{(1)}) \downarrow$	$\mathcal{E}(v^{(2)}) \downarrow$	$\mathcal{E}(v^{(3)}) \downarrow$	$\mathcal{E}(v) \downarrow$
DSM (Nelsonian)	$\mathbf{4.441 \times 10^{-5}}$	$\mathbf{2.721 \times 10^{-5}}$	2.810×10^{-5}	$\mathbf{3.324 \times 10^{-5}}$
DSM (Gradient Divergence)	6.648×10^{-5}	4.405×10^{-5}	$\mathbf{1.915 \times 10^{-5}}$	4.333×10^{-5}



(a) The Nelsonian version.



(b) The Gradient Divergence version.

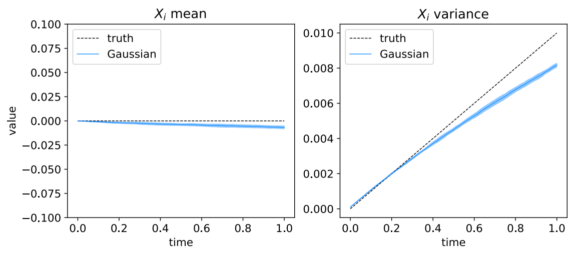
Figure C.3: The density function for 3d harmonic oscillator using two versions of the proposed approach.

Table C.2: 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 (4.14). The **best** result is in bold. These results demonstrate that our approach work better than the naíve sampling scheme.

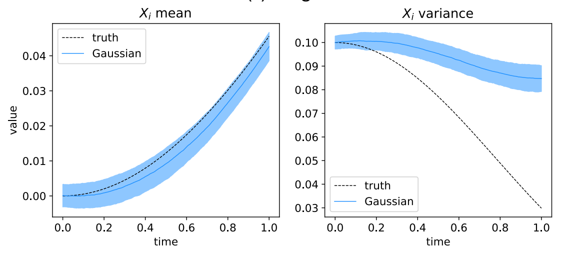
Problem	Model	$\mathcal{E}_m(X_i)\downarrow$	$\mathcal{E}_v(X_i)\downarrow$	$\mathcal{E}(v)\downarrow$	$\mathcal{E}(u)\downarrow$
Singular IC	Gaussian sampling	0.043 ± 0.042	0.146 ± 0.013	1.262	0.035
	DSM	**0.008 ± 0.007**	**0.011 ± 0.007**	**0.524**	**0.008**
Harm osc 1d,	Gaussian sampling	0.294 ± 0.152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

C.4.4 Scaling experiments for non-interacting system

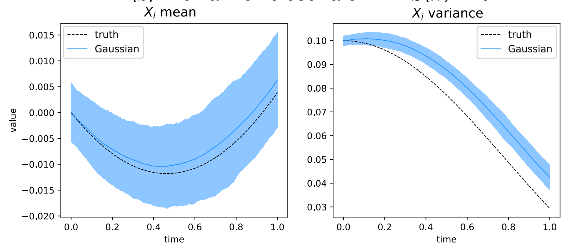
We empirically estimate memory allocation on a GPU (NVIDIA A100) when training two versions of the proposed algorithm. In addition, we estimate the number of epochs until the training loss function is less than 10^{-2} for different problem dimensions. Figure C.5(a)



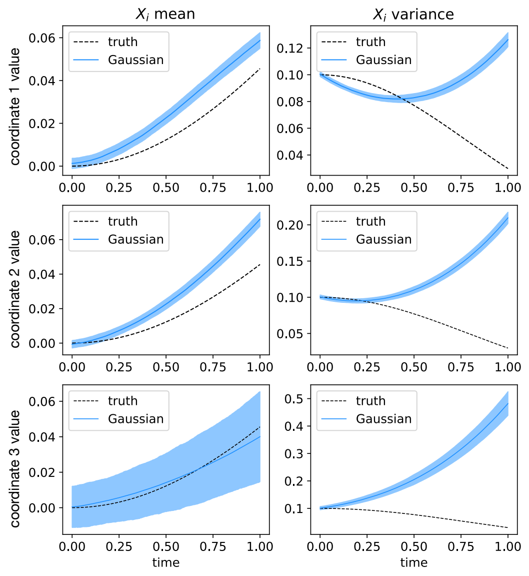
(a) Singular IC



(b) The harmonic oscillator with $S(x) \equiv 0$



(c) The harmonic oscillator with $S(x) = -5x$

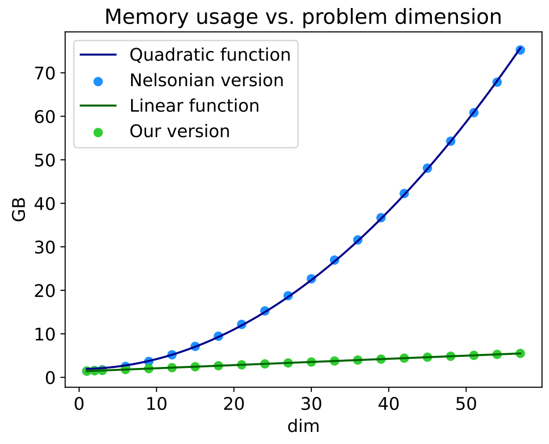


(d) The harmonic oscillator with $S(x) = 0$ in 3d

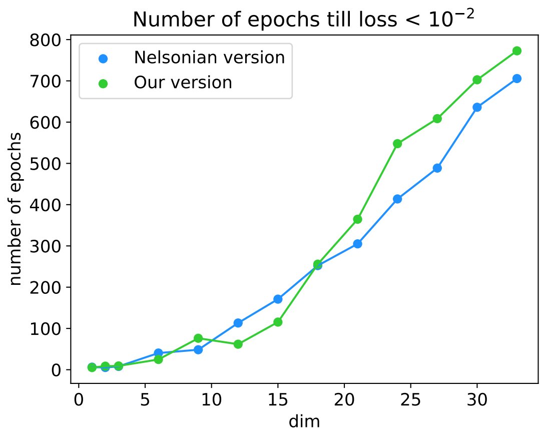
Figure C.4: An illustration of produced trajectories using the naïve Gaussian sampling scheme as a comparison with the proposed approach. The obtained trajectories do not match the solution, while the results in our paper suggest that the proposed DSM approach converges better. Compare with Figures 4.2, C.1, C.2.

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 C.5(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 4.3.4.

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 4.3.4, and the training



(a) GPU memory usage.



(b) Number of epochs until loss $< 10^{-2}$.

Figure C.5: Empirical complexity evaluation of two versions of the proposed method: memory usage and the number of epochs until the training loss is less than the threshold.

time grows linearly with the problem dimension. Table C.3 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 C.3: Training time and test errors for the harmonic oscillator model for different d.

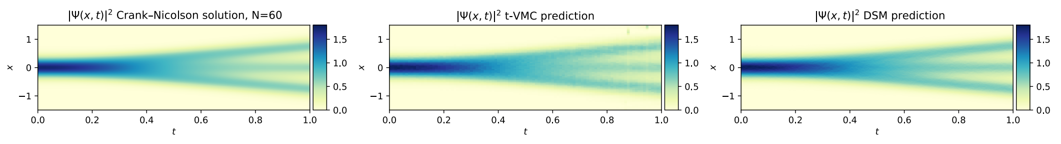
d	$\mathcal{E}_m(X_i) \downarrow$	$\mathcal{E}_v(X_i) \downarrow$	$\mathcal{E}(v) \downarrow$	$\mathcal{E}(u) \downarrow$	Train time
1	0.074 ± 0.052	0.009 ± 0.007	0.00012	2.809e-05	46m 20s
3	0.073 ± 0.048	0.010 ± 0.008	4.479×10^{-5}	3.946×10^{-5}	2h 18m
5	0.081 ± 0.057	0.009 ± 0.008	4.956×10^{-5}	4.000×10^{-5}	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

C.4.5 Scaling experiments for the interacting system

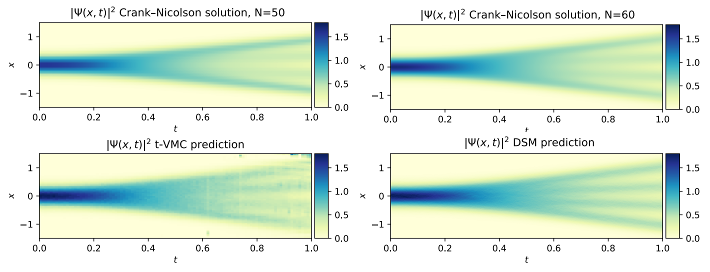
This section provides more details on experiments from Section 4.3.4, where we investigate the scaling of the DSM approach for the interacting bosons system. We compare the performance of our algorithm with a numerical solver based on the Crank–Nicolson method (we modified the QMSOLVE library to work 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 4.3 and Table 4.4.

Figure C.6 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 4.3.3. 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.

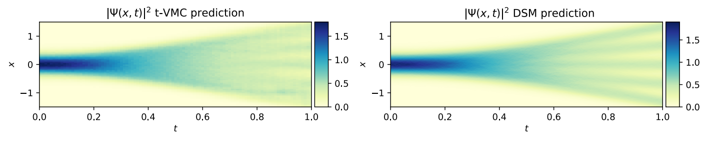
As for the DSM implementation details, we fix hyperparameters and only change d: for example, the neural network size is 500, and the batch size is 100. We train our method until the average training loss becomes lower than a particular threshold (0.007). These numbers are reported for a GPU A40. The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



a) Three particles



b) Four particles



c) Five particles

Figure C.6: Probability density plots for different numbers of interacting particles d. For five particles, our computer system does not allow running the Crank-Nicolson solver.

C.4.6 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.3. All models are trained for an equal number of epochs across every hyper-parameter setting, and the results are presented in Figure C.7. 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.

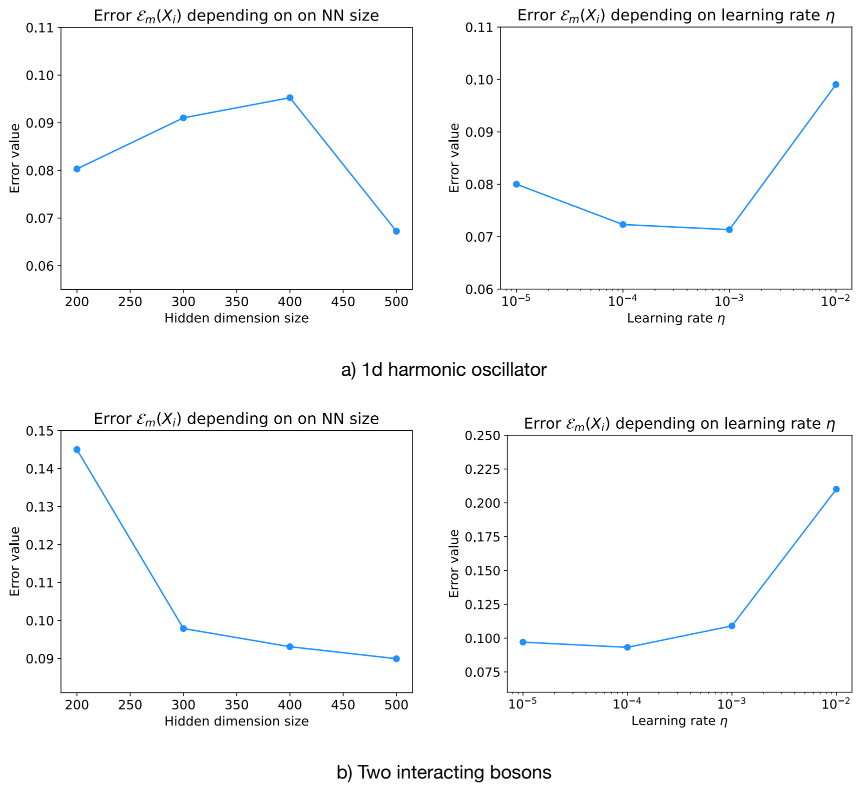


Figure C.7: Sensitivity analysis of the neural network hyperparameters for the proposed method on two systems: (a) a 1D harmonic oscillator with $S_0(x) \equiv 0$, and (b) a system of two interacting bosons. The plots illustrate the impact of varying the hidden layer size and the learning rate on the model's performance, quantified by the $\mathcal{E}_m(X_i)$ error metric.

C.5 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 [1966a].

C.5.1 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:

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

Dividing by ψ[3], and separating real and imaginary parts, we obtain

$$-\hbar\partial_t S = -\frac{\hbar^2}{2m}\Big(\frac{1}{2}\Delta\log\rho + \frac{1}{4}\|\log\rho\|^2 - \|\nabla S\|^2\Big) + V, \tag{C.2}$$
$$\frac{\hbar}{2}\partial_t\log\rho = -\frac{\hbar^2}{2m}\big(\Delta S + \langle\log\rho, \nabla S\rangle\big). \tag{C.3}$$

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

$$m\frac{\hbar}{m}\partial_t S = \frac{\hbar}{2m}\langle\nabla, u\rangle + \frac{1}{2}\|u\|^2 - \frac{1}{2}\|v\|^2 - V, \tag{C.4}$$
$$\frac{\hbar}{2m}\partial_t\log\rho = -\frac{\hbar}{2m}\langle\nabla, v\rangle - \langle u, v\rangle. \tag{C.5}$$

3. 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.

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

$$\partial_t v = -\frac{1}{m}\nabla V + \langle u, \nabla\rangle u - \langle v, \nabla\rangle v + \frac{\hbar}{2m}\nabla\langle\nabla, u\rangle, \tag{C.6}$$

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

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

$$v(x, t) = \frac{\hbar}{m}\nabla S(x, t), \tag{C.8}$$

$$u(x, t) = \frac{\hbar}{2m}\nabla \log \rho(x, t) \tag{C.9}$$

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 [1966a, 2005], Guerra [1995]. Moreover, this equivalence holds for manifolds $\mathcal{M}$ with trivial second cohomology group as noted in Alvarez [1986], Wallstrom [1989], Prieto and Vitolo [2014].

C.5.2 Novel equations of stochastic mechanics

We note that our equations differ from Guerra [1995], Nelson [1966a]. In Nelson [1966a], we see

$$\partial_t v = -\frac{1}{m}\nabla V + \langle u, \nabla\rangle u - \langle v, \nabla\rangle v + \frac{\hbar}{2m}\Delta u, \tag{C.10a}$$

$$\partial_t u = -\nabla\langle v, u\rangle - \frac{\hbar}{2m}\nabla\langle\nabla, v\rangle; \tag{C.10b}$$

and in Guerra [1995], we see

$$\partial_t v = -\frac{1}{m}\nabla V + \langle u, \nabla\rangle u - \langle v, \nabla\rangle v + \frac{\hbar}{2m}\Delta u, \tag{C.11a}$$

$$\partial_t u = -\nabla\langle v, u\rangle - \frac{\hbar}{2m}\Delta v. \tag{C.11b}$$

Note that our equations (4.7a), (4.7b) do not directly use the second-order Laplacian operator Δ, as it appears for u in Equation (C.10a) and v in Equation (C.11b). 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 [1966a] 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.

C.5.3 Diffusion processes of stochastic mechanics

Let's consider an arbitrary Ito diffusion process

$$\mathrm{d}X(t) = b(X(t), t)\mathrm{d}t + \sigma(X(t), t)\mathrm{d}\overrightarrow{W}, \tag{C.12}$$

$$X(0) \sim \rho_0, \tag{C.13}$$

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:

$$\partial_t \rho_X = \langle \nabla, b\rho_X \rangle + \frac{1}{2}\mathrm{Tr}\big(\nabla^2 \cdot (\sigma\sigma^T \rho_X)\big). \tag{C.14}$$

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

$$\partial_t \rho_X = \langle \nabla, b^* \rho_X \rangle - \frac{1}{2}\mathrm{Tr}\big(\nabla^2 \cdot (\sigma\sigma^T \rho_X)\big), \tag{C.15}$$

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

$$\partial_t \rho_X = \langle \nabla, v\rho_X \rangle, \tag{C.16}$$

where $v = \dfrac{b+b^*}{2}$ is so called probability current. This is the continuity equation for the Ito diffusion process from Equation (C.12). We refer to Anderson [1982] for details. We note that the same Equation (C.16) can be obtained with an arbitrary non-singular $\sigma(x,t)$ as long as $v = v(x,t)$ remains fixed.

Proposition C.5.1. *Consider arbitrary $\nu > 0$, denote $\rho = |\psi|^2$ and consider decomposition $\psi = \sqrt{\rho}e^{iS}$. Then the following process $X(t)$:*

$$\mathrm{d}X(t) = \big(\nabla S(X(t),t) + \frac{\nu\hbar}{2m}\nabla \log \rho(X(t),t)\big)\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\vec{W}, \tag{C.17}$$

$$X(0) \sim |\psi_0|^2, \tag{C.18}$$

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:

$$i\hbar\partial_t\psi = (-\frac{\hbar^2}{2m}\Delta + V)\psi, \tag{C.19}$$

$$\psi(\cdot,0) = \psi_0 \tag{C.20}$$

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:

$$\partial_t\rho = \langle\nabla, v\rho\rangle, \tag{C.21}$$

$$\rho(\cdot,0) = \big|\psi_0\big|^2. \tag{C.22}$$

This immediately tells us what should be initial distribution ρ_0 and $\frac{b+b^*}{2}$ for the Ito diffusion process from Equation (C.12).

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 $u(x,t) = \frac{\hbar}{2m}\nabla \log \rho(x,t)$ and using arbitrary $\nu > 0$ we derive

$$\partial_t \rho = \langle \nabla, (v + \nu u)\rho\rangle + \frac{\nu\hbar}{2m}\Delta\rho. \tag{C.23}$$

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 $\sigma(x,t) \equiv \frac{\nu\hbar}{m} I_d$:

$$\mathrm{d}X(t) = (v(X(t),t) + \nu u(X(t),t))\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\vec{W}, \tag{C.24}$$

$$X(0) \sim |\psi_0|^2. \tag{C.25}$$

□

To obtain numerical samples from the diffusion, one can use any numerical integrator, for example, the Euler-Maruyama integrator [Kloeden and Platen, 1992]:

$$X_{i+1} = X_i + (v(X_i,t_i) + \nu u(X_i,t_i))\epsilon + \sqrt{\frac{\nu\hbar}{m}}\epsilon\mathcal{N}(0, I_d), \tag{C.26}$$

$$X_0 \sim |\psi_0|^2, \tag{C.27}$$

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 and Platen, 1992], can achieve the same integration error with larger $\epsilon > 0$; this approach is out of the scope of our work.

C.5.4 Interpolation between Bohmian and Nelsonian pictures

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

$$\mathrm{d}X(t) = (v(X(t),t) + \nu u(X(t),t))\mathrm{d}t + \sqrt{\frac{\nu\hbar}{m}}\mathrm{d}\vec{W}, \tag{C.28}$$

$$X(0) \sim |\psi_0|^2 \tag{C.29}$$

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

$$\mathrm{d}X(t) = (v(X(t),t) + u(X(t),t))\mathrm{d}t + \sqrt{\frac{\hbar}{m}}\mathrm{d}\vec{W}, \tag{C.30}$$

$$X(0) \sim |\psi_0|^2 \tag{C.31}$$

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:

$$\mathrm{d}X(t) = v(X(t),t)\mathrm{d}t, \tag{C.32}$$

$$X(0) \sim |\psi_0|^2. \tag{C.33}$$

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 and Struyve, 2010]. Though, under certain mild conditions [Boffi and 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.

C.5.5 Computational complexity

Proposition C.5.2 (Remark 4.2.1)**.** *The algorithmic complexity w.r.t. d of computing differential operators from Equations (4.8), (4.9) 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 (4.8), (4.9) 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 and 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.

C.6 On strong convergence

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

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:

$$\mathrm{d}X(t) = \big(\widetilde{v}(X(t),t)+\widetilde{u}(X(t),t)\big)\mathrm{d}t + \sqrt{\frac{\hbar}{m}}\mathrm{d}\overrightarrow{W}^X(t), \tag{C.34}$$

$$\mathrm{d}Y(t) = \big(v(Y(t),t)+u(Y(t),t)\big)\mathrm{d}t + \sqrt{\frac{\hbar}{m}}\mathrm{d}\overrightarrow{W}^Y(t), \tag{C.35}$$

$$X(0)\sim|\psi_0|^2, \tag{C.36}$$

$$Y(0) = X(0), \tag{C.37}$$

where u,v are true solutions to Equation (C.10). 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 $\frac{\mathrm{d}\left[\overrightarrow{W}^X,\overrightarrow{W}^Y\right]_t}{\mathrm{d}t} = \lim_{\mathrm{d}t\to0_+}\mathbb{E}\Big(\frac{\left(\overrightarrow{W}^X(t+\mathrm{d}t)-\overrightarrow{W}^X(t)\right)\left(\overrightarrow{W}^Y(t+\mathrm{d}t)-\overrightarrow{W}^Y(t)\right)}{\mathrm{d}t}\Big|\overrightarrow{\mathcal{F}}_t\Big)$. We specify it as $\mathrm{d}\left[\overrightarrow{W}^X,\overrightarrow{W}^Y\right]_t = I_d\mathrm{d}t$, 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:

$$L_1(\widetilde{v},\widetilde{u}) = \int_0^T \mathbb{E}^X\big\|\partial_t\widetilde{u}(X(t),t) - \mathcal{D}_u[\widetilde{v},\widetilde{u},x,t]\big\|^2\mathrm{d}t, \tag{C.38}$$

$$L_2(\widetilde{v},\widetilde{u}) = \int_0^T \mathbb{E}^X\big\|\partial_t\widetilde{v}(X(t),t) - \mathcal{D}_v[\widetilde{v},\widetilde{u},X(t),t]\big\|^2\mathrm{d}t, \tag{C.39}$$

$$L_3(\widetilde{u},\widetilde{v}) = \mathbb{E}^X\|\widetilde{u}(X(0),0)-u(X(0),0)\|^2, \tag{C.40}$$

$$L_4(\widetilde{u},\widetilde{v}) = \mathbb{E}^X\|\widetilde{v}(X(0),0)-v(X(0),0)\|^2. \tag{C.41}$$

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

C.6.1 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 $\overrightarrow{\mathcal{F}_t}$:

$$\begin{aligned} \mathrm{d}Z(t) &= F(t)\mathrm{d}t + G(t)\mathrm{d}\overrightarrow{W}, \\ Z(0) &\sim \rho_0. \end{aligned} \tag{C.42}$$

Moreover, assume $\left[Z(t), Z(t)\right]_t = \mathbb{E}\int_0^t G^T G(t)\mathrm{d}t < \infty$, $\mathbb{E}\int_0^t \|F(t)\|^2 \mathrm{d}t < \infty$. 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

$$\overrightarrow{M}^f(t) = f(Z(t), t) - f(Z(0), 0) - \int_0^t \overrightarrow{\mathcal{L}}^X f(Z(t), t) \text{ is } \overrightarrow{\mathcal{F}_t}\text{-martingale.} \tag{C.43}$$

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:

$$\overleftarrow{M}^f(t) = f(Z(t), t) - f(Z(0), 0) - \int_0^t \overleftarrow{\mathcal{L}}^X f(Z(t), t) \text{ is } \overleftarrow{\mathcal{F}_t}\text{-martingale} \tag{C.44}$$

For more information on the properties of generators, we refer to Baldi and Baldi [2017].

Lemma C.6.1. *(Itô Lemma, [Baldi and Baldi, 2017, Theorem 8.1 and Remark 9.1])*

$$\overrightarrow{\mathcal{L}}^Z f(x,t) = \partial_t f(x,t) + \langle \nabla f(x,t), F(t)\rangle + \frac{\hbar}{2m}\mathrm{Tr}\big(G^T(t)\nabla^2 f(x,s) G(t)\big). \tag{C.45}$$

Lemma C.6.2. *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*

$$\overleftarrow{\mathcal{L}}^Z f(x,t) = \partial_t f(x,t) + \langle \nabla f(x,t), F(t) - \frac{\hbar}{m}\nabla \log p_Z(x,t)\rangle - \frac{1}{2}\mathrm{Tr}\big(G^T(t)\nabla^2 f(x,s)G(t)\big). \tag{C.46}$$

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 C.6.1 and grouping all terms yields the statement. □

Lemma C.6.3. *The following identity holds for any process* $Z(t)$*:*

$$\overrightarrow{\mathcal{L}}^Z \overleftarrow{\mathcal{L}}^Z x = \overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z x. \tag{C.47}$$

Proof. One needs to recognize that Equation (C.16) is the difference between two types of generators, we automatically have the following identity that holds for any process Z. □

Lemma C.6.4. *(Nelson's Lemma, Nelson [2020a])*

$$\begin{aligned}&\mathbb{E}^Z\Big(f(Z(t),t)g(Z(t),t) - f(Z(0),t)g(Z(0),t)\Big)\\ &\quad= \mathbb{E}^Z\int_0^t \Big(\overrightarrow{\mathcal{L}}^Z f(Z(s),t)g(Z(s),t) + f(Z(s),t)\overleftarrow{\mathcal{L}}^Z g(Z(s),s)\Big)\mathrm{ds}\end{aligned} \tag{C.48}$$

Lemma C.6.5. *It holds that:*

$$\mathbb{E}^Z\Big(\|Z(t)\|^2 - \|Z(0)\|^2\Big) \tag{C.49}$$

$$= \int_0^t \mathbb{E}^Z\Big(2\langle \overleftarrow{\mathcal{L}}^Z Z(0), Z(s)\rangle + 2\int_0^s \langle \overleftarrow{\mathcal{L}}^Z\overrightarrow{\mathcal{L}}^Z Z(z), Z(s)\rangle \mathrm{d}z\Big)\mathrm{ds} + \big[Z(t),Z(t)\big]_t \tag{C.50}$$

Proof. By using Itô Lemma C.6.1 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 $\mathcal{F}_{\overleftarrow{s'}}$-martingale for $s' \le s$, 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.

□

C.6.2 Adjoint processes

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

$$\mathrm{d}X'(t) = \big(\widetilde{v}(X'(t), t) + \widetilde{u}(X'(t), t)\big)\mathrm{d}t + \sqrt{\frac{\hbar}{2m}}\mathrm{d}\overleftarrow{W}^X(t). \tag{C.51}$$

We call such process as adjoint to the process X. Lemma C.6.3 can be generalized to the pair of adjoint processes (X, X') in the following way and will be instrumental in proving our results.

Lemma C.6.6. *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 $\widetilde{v} - \widetilde{u}$:*

$$\overrightarrow{\mathcal{L}}^X \overleftarrow{\mathcal{L}}^{X'} x - \overleftarrow{\mathcal{L}}^{X'} \overrightarrow{\mathcal{L}}^X x = \overleftarrow{\mathcal{L}}^{X'} \overleftarrow{\mathcal{L}}^{X'} x - \overrightarrow{\mathcal{L}}^X \overrightarrow{\mathcal{L}}^X x. \tag{C.52}$$

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 (4.7b) for both cases. This equation is zero only if $\widetilde{u} = \frac{\hbar}{2m}\nabla \log p_X$ □

Lemma C.6.7. *The following bound holds:*

$$\left\|\big(\overrightarrow{\mathcal{L}}^X + \overleftarrow{\mathcal{L}}^X\big)(\widetilde{u} - \frac{\hbar}{2m}\nabla \log p_X)\right\| \leq \left\|\overrightarrow{\mathcal{L}}^X \overleftarrow{\mathcal{L}}^{X'} x - \overleftarrow{\mathcal{L}}^{X'} \overrightarrow{\mathcal{L}}^X x\right\| + 2\|\nabla \widetilde{v}\|_\infty \left\|\widetilde{u} - \frac{\hbar}{2m}\nabla \log p_X\right\|. \tag{C.53}$$

Proof. First, using Lemma C.6.6 we obtain:

$$\begin{aligned}
& \overrightarrow{\mathcal{L}}^X \overleftarrow{\mathcal{L}}^X x - \overleftarrow{\mathcal{L}}^X \overrightarrow{\mathcal{L}}^X x = 0 \\
\iff & \overrightarrow{\mathcal{L}}^X\big(\widetilde{v} + \widetilde{u} - \frac{\hbar}{m}\nabla \log p_X\big) - \overleftarrow{\mathcal{L}}^X\big(\widetilde{v} + \widetilde{u}\big) = 0 \\
\iff & \overrightarrow{\mathcal{L}}^X\big((\widetilde{v} - \widetilde{u}) + (2\widetilde{u} - \frac{\hbar}{m}\nabla \log p_X)\big) - \overleftarrow{\mathcal{L}}^X\big(\widetilde{v} + \widetilde{u}\big) = 0 \\
\iff & \overrightarrow{\mathcal{L}}^X\big((\widetilde{v} - \widetilde{u}) + (2\widetilde{u} - \frac{\hbar}{m}\nabla \log p_X)\big) - \overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v} + \widetilde{u}\big) + \Big(\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v} + \widetilde{u}\big) - \overleftarrow{\mathcal{L}}^X\big(\widetilde{v} + \widetilde{u}\big)\Big) = 0 \\
\iff & \overrightarrow{\mathcal{L}}^X\big(2\widetilde{u} - \frac{\hbar}{m}\nabla \log p_X\big) + \overrightarrow{\mathcal{L}}^X(\widetilde{v} - \widetilde{u}) - \overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v} + \widetilde{u}\big) + \Big(\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v} + \widetilde{u}\big) - \overleftarrow{\mathcal{L}}^X\big(\widetilde{v} + \widetilde{u}\big)\Big) = 0.
\end{aligned} \tag{C.54}$$

Then, we note that:

$$\overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v} + \widetilde{u}\big) - \overleftarrow{\mathcal{L}}^X\big(\widetilde{v} + \widetilde{u}\big) = \langle \frac{\hbar}{m}\nabla \log p_X - 2\widetilde{u}, \nabla(\widetilde{v} + \widetilde{u})\rangle. \tag{C.55}$$

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 C.6.6 to time-reversal X' we obtain:

$$\begin{aligned}
&\overleftarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^X x - \overrightarrow{\mathcal{L}}^X\overrightarrow{\mathcal{L}}^X x = 0\\
\iff\ & \overleftarrow{\mathcal{L}}^X\big(\widetilde{v}+\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big) - \overrightarrow{\mathcal{L}}^X\big(\widetilde{v}+\widetilde{u}\big) = 0\\
\iff\ & \overleftarrow{\mathcal{L}}^X\big((\widetilde{v}-\widetilde{u})+(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X)\big) - \overrightarrow{\mathcal{L}}^X\big(\widetilde{v}+\widetilde{u}\big) = 0\\
\iff\ & \overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}-\widetilde{u}\big) + \overleftarrow{\mathcal{L}}^X\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big) - \overrightarrow{\mathcal{L}}^X\big(\widetilde{v}+\widetilde{u}\big) + \Big(\overleftarrow{\mathcal{L}}^X\big(\widetilde{v}-\widetilde{u}\big) - \overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}-\widetilde{u}\big)\Big) = 0\\
\iff\ & \overleftarrow{\mathcal{L}}^X\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big) + \overleftarrow{\mathcal{L}}^{X'}\big(\widetilde{v}-\widetilde{u}\big) - \overrightarrow{\mathcal{L}}^X\big(\widetilde{v}+\widetilde{u}\big) - \langle\frac{\hbar}{m}\nabla\log p_X - 2\widetilde{u},\nabla(\widetilde{v}-\widetilde{u})\rangle = 0\\
\iff\ & \overleftarrow{\mathcal{L}}^X\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big) + \overleftarrow{\mathcal{L}}^{X'}\overleftarrow{\mathcal{L}}^{X'}x - \overrightarrow{\mathcal{L}}^X\overrightarrow{\mathcal{L}}^X x - \langle\frac{\hbar}{m}\nabla\log p_X - 2\widetilde{u},\nabla(\widetilde{v}-\widetilde{u})\rangle = 0.
\end{aligned} \tag{C.56}$$

By using Lemma C.6.6 we thus derive:

$$\overleftarrow{\mathcal{L}}^X\big(2\widetilde{u}-\frac{\hbar}{m}\nabla\log p_X\big) + \overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}x - \overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^X x - \langle\frac{\hbar}{m}\nabla\log p_X - 2\widetilde{u},\nabla(\widetilde{v}-\widetilde{u})\rangle = 0. \tag{C.57}$$

Summing up both identities, therefore, yields:

$$\Big(\overleftarrow{\mathcal{L}}^X+\overrightarrow{\mathcal{L}}^X\Big)\big(\widetilde{u}-\frac{\hbar}{2m}\nabla\log p_X\big) + \overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}x - \overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^X x + 2\langle\widetilde{u}-\frac{\hbar}{2m}\nabla\log p_X,\nabla\widetilde{v}\rangle = 0. \tag{C.58}$$

□

Theorem C.6.8. *The following bound holds:*

$$\sup_{0\le t\le T}\mathbb{E}^X\big\|\widetilde{u}(X(t),t)-\frac{\hbar}{2m}\nabla\log p_X(X(t),t)\big\|^2 \le e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\big(L_3(\widetilde{v},\widetilde{u})+L_2(\widetilde{v},\widetilde{u})\big). \tag{C.59}$$

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

C.6.4, we have the following identity:

$$\begin{aligned}
&\mathbb{E}^X\|\widetilde{u}(X(t),t)-\frac{\hbar}{2m}\nabla\log p_X(X(t),t)\|^2-\mathbb{E}^X\|\widetilde{u}(X(0),0)-\frac{\hbar}{2m}\nabla\log p_X(X(0),0)\|^2\\
&=\mathbb{E}^X\int_0^t\langle u(X(s),s)-\frac{\hbar}{2m}\nabla\log p_X(X(s),s),\\
&\qquad\big(\overrightarrow{\mathcal{L}}^X+\overleftarrow{\mathcal{L}}^X\big)\big(u(X(s),s)-\frac{\hbar}{2m}\nabla\log p_X(X(s),s)\big)\rangle\mathrm{d}s.
\end{aligned}\tag{C.60}$$

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 $\langle a,b\rangle\le\frac{1}{2}\big(\|a\|^2+\|b\|^2\big)$ we obtain:

$$\begin{aligned}
&\mathbb{E}^X\|u(X(t),t)-\frac{\hbar}{2m}\nabla\log p_X(X(t),t)\|^2-L_3(\widetilde{v},\widetilde{u})\\
&\le\int_0^t\Big(\frac{1}{2}\mathbb{E}^X\|u(X(s),s)-\frac{\hbar}{2m}\nabla\log p_X(X(s),s)\|^2\\
&\quad+\frac{1}{2}\mathbb{E}^X\Big\|\big(\overrightarrow{\mathcal{L}}^X+\overleftarrow{\mathcal{L}}^X\big)\big(u(X(s),s)-\frac{\hbar}{2m}\nabla\log p_X(X(s),s)\big)\Big\|^2\Big)\mathrm{d}s
\end{aligned}\tag{C.61}$$

Using Lemma C.6.7, we obtain:

$$\begin{aligned}
&\mathbb{E}^X\|u(X(t),t)-\frac{\hbar}{2m}\nabla\log p_X(X(t),t)\|^2-L_3(\widetilde{v},\widetilde{u})\\
&\le\int_0^t\Big(\frac{1}{2}\mathbb{E}^X\|u(X(s),s)-\frac{\hbar}{2m}\nabla\log p_X(X(s),s)\|^2\\
&\quad+\Big\|\overrightarrow{\mathcal{L}}^X\overleftarrow{\mathcal{L}}^{X'}x-\overleftarrow{\mathcal{L}}^{X'}\overrightarrow{\mathcal{L}}^X x\Big\|^2+4\|\nabla\widetilde{v}\|_\infty^2\Big\|\widetilde{u}-\frac{\hbar}{2m}\nabla\log p_X\Big\|^2\Big)\mathrm{d}s
\end{aligned}\tag{C.62}$$

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

$$\begin{aligned}
&\mathbb{E}^X\|u(X(t),t)-\frac{\hbar}{2m}\nabla\log p_X(X(t),t)\|^2\\
&\quad\le L_3(\widetilde{v},\widetilde{u})+L_2(\widetilde{v},\widetilde{u})+\int_0^t\big(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\big)\mathbb{E}^X\|u(X(s),s)-\frac{\hbar}{2m}\nabla\log p_X(X(s),s)\|^2\mathrm{d}s.
\end{aligned}\tag{C.63}$$

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

Considering those two operators, we can rewrite the equations (C.10) alternatively as:

$$\frac{1}{2}\Big(\overset{\rightarrow}{\mathcal{L}}^{Y}\overset{\leftarrow}{\mathcal{L}}^{Y}x+\overset{\leftarrow}{\mathcal{L}}^{Y}\overset{\rightarrow}{\mathcal{L}}^{Y}x\Big)=-\frac{1}{m}\nabla V(x), \tag{C.64}$$

$$\frac{1}{2}\Big(\overset{\rightarrow}{\mathcal{L}}^{Y}\overset{\leftarrow}{\mathcal{L}}^{Y}x-\overset{\leftarrow}{\mathcal{L}}^{Y}\overset{\rightarrow}{\mathcal{L}}^{Y}x\Big)=0. \tag{C.65}$$

This leads us to the identity:

$$\overset{\rightarrow}{\mathcal{L}}^{Y}\overset{\leftarrow}{\mathcal{L}}^{Y}x=-\frac{1}{m}\nabla V(x). \tag{C.66}$$

Lemma C.6.9. *We have the following bound:*

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

Proof. Consider rewriting losses as:

$$L_1(\widetilde{v},\widetilde{u})=\int_0^t \mathbb{E}_{t\sim U[0,T]}\mathbb{E}^X\Big\|\frac{1}{2}\big(\overset{\rightarrow}{\mathcal{L}}^{X}\overset{\leftarrow}{\mathcal{L}}^{X'}X(t)+\overset{\rightarrow}{\mathcal{L}}^{X}\overset{\leftarrow}{\mathcal{L}}^{X'}X(t)\big)+\frac{1}{m}\nabla V(X(t))\Big\|^2\mathrm{d}t, \tag{C.67}$$

$$L_2(\widetilde{v},\widetilde{u})=\frac{1}{4}\int_0^t \mathbb{E}_{t\sim U[0,T]}\mathbb{E}^X\Big\|\overset{\rightarrow}{\mathcal{L}}^{X}\overset{\leftarrow}{\mathcal{L}}^{X'}X(t)-\overset{\rightarrow}{\mathcal{L}}^{X'}\overset{\leftarrow}{\mathcal{L}}^{X}X(t)\Big\|^2\mathrm{d}t. \tag{C.68}$$

Using the triangle inequality yields the statement. □

Lemma C.6.10. *We have the following bound:*

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

Proof. From (C.55) we have:

$$\overleftarrow{\mathcal{L}}^X \overrightarrow{\mathcal{L}}^X X(t) = \overleftarrow{\mathcal{L}}^{X'} \overrightarrow{\mathcal{L}}^X X(t) + \langle \frac{\hbar}{m}\nabla \log p_X - 2\widetilde{u}, \nabla(\widetilde{v}+\widetilde{u})\rangle. \tag{C.69}$$

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

$$\int_0^t \mathbb{E}^X \Big\|\overleftarrow{\mathcal{L}}^X \overrightarrow{\mathcal{L}}^X X(t) + \frac{1}{m}\nabla V(X(t))\Big\|^2 \mathrm{d}t \tag{C.70}$$

$$\le 2\big(\|\widetilde{u}\|_\infty + \|\widetilde{v}\|_\infty\big)^2 \int_0^t \mathbb{E}^X \Big\|u(X(t),t) - \frac{\hbar}{2m}\log p_X(X(t),t)\Big\|^2 \mathrm{d}t + 4L_1(\widetilde{v},\widetilde{u}) + 4L_2(\widetilde{v},\widetilde{u}). \tag{C.71}$$

Using Theorem C.6.8 concludes the proof. □

Lemma C.6.11. *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:*

$$\overrightarrow{\mathcal{L}}^Z h = \overrightarrow{\mathcal{L}}^X f + \overrightarrow{\mathcal{L}}^Y g \tag{C.72}$$

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

$$\overrightarrow{\mathcal{L}}^Z f = \overrightarrow{\mathcal{L}}^X f \tag{C.73}$$

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 C.6.12. *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 C.6.13. *(Strong Convergence) Let the loss be defined as $\mathcal{L}(\widetilde{v}, \widetilde{u}) = \sum_{i=1}^4 w_i L_i(\widetilde{v}, \widetilde{u})$ for some arbitrary constants $w_i > 0$. Then we have the following bound between processes X and Y:*

$$\sup_{t \le T} \mathbb{E}\|X(t) - Y(t)\|^2 \le C_T \mathcal{L}(\widetilde{v}, \widetilde{u}) \tag{C.74}$$

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

Proof. We are going to prove the bound:

$$\sup_{t \le T} \mathbb{E}\|X(t) - Y(t)\|^2 \le \sum_{i=1}^4 w_i' L_i(\widetilde{v}, \widetilde{u}) \tag{C.75}$$

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

$$\sum_{i=1}^4 w_i' L_i(\widetilde{v}, \widetilde{u}) \le \sum_{i=1}^4 \frac{w_i'}{w_i} w_i L_i(\widetilde{v}, \widetilde{u}) \le \big(\max_i \frac{w_i'}{w_i}\big) \sum_{i=1}^4 w_i L_i(\widetilde{v}, \widetilde{u}) = C_T \mathcal{L}(\widetilde{v}, \widetilde{u}) \tag{C.76}$$

First, we apply Lemma C.6.5 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:

$$\begin{aligned}
&\mathbb{E}^Z \|X(t) - Y(t)\|^2 \\
&= \int_0^t \mathbb{E}^Z \Big(2\langle \overleftarrow{\mathcal{L}}^Z (X(0) - Y(0)), X(s) - Y(s)\rangle \\
&\quad + 2\int_0^s \langle \overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z (X(z) - Y(z)), X(s) - Y(s)\rangle \mathrm{d}z \Big) \mathrm{d}s \\
&\leq \int_0^t \mathbb{E}^Z \Big(\big\|\overleftarrow{\mathcal{L}}^Z (X(0) - Y(0))\big\|^2 + \|X(s) - Y(s)\|^2 \\
&\quad + \int_0^s \Big(\big\|\overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z (X(z) - Y(z))\big\|^2 + \|X(s) - Y(s)\|^2 \mathrm{d}z \Big)\Big) \mathrm{d}s \\
&\leq \int_0^t \mathbb{E}^Z \Big(\big\|\overleftarrow{\mathcal{L}}^Z (X(0) - Y(0))\big\|^2 + (1+T)\|X(s) - Y(s)\|^2 \\
&\quad + \int_0^s \big\|\overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z (X(z) - Y(z))\big\|^2 \mathrm{d}z \Big) \mathrm{d}s.
\end{aligned} \tag{C.77}$$

Then, using Corollary C.6.12, (C.66) and then Lemma C.6.10 we obtain that

$$\begin{aligned}
\int_0^s \big\|\overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z (X(z) - Y(z))\big\|^2 \mathrm{d}z &= \int_0^s \big\|\overleftarrow{\mathcal{L}}^Z \overrightarrow{\mathcal{L}}^Z X(z) + \frac{1}{m}\nabla V(X(z))\big\|^2 \mathrm{d}z \\
&\leq 2T\big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\big(L_3(\widetilde{v},\widetilde{u}) + L_2(\widetilde{v},\widetilde{u})\big) + 4L_1(\widetilde{v},\widetilde{u}) + 4L_2(\widetilde{v},\widetilde{u}).
\end{aligned} \tag{C.78}$$

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

$$\int_0^t \mathbb{E}^Z \Big(\big\|\overleftarrow{\mathcal{L}}^Z (X(0) - Y(0))\big\|^2 \leq 2TL_3(\widetilde{v},\widetilde{u}) + 2TL_4(\widetilde{v},\widetilde{u}), \tag{C.79}$$

where we used triangle inequality. Combining obtained bounds yields:

$$\begin{aligned}
&\mathbb{E}^Z\|X(t)-Y(t)\|^2\\
&\leq \int_0^t (1+T)\|X(s)-Y(s)\|^2 \mathrm{d}s\\
&\quad + 2TL_3(\widetilde{v},\widetilde{u}) + 2TL_4(\widetilde{v},\widetilde{u})\\
&\quad + 2T\big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\big(L_3(\widetilde{v},\widetilde{u}) + L_2(\widetilde{v},\widetilde{u})\big)\\
&\quad + 4L_1(\widetilde{v},\widetilde{u}) + 4L_2(\widetilde{v},\widetilde{u})\\
&= \int_0^t (1+T)\|X(s)-Y(s)\|^2 \mathrm{d}s\\
&\quad + 4L_1(\widetilde{v},\widetilde{u}) + \Big(2T\big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T} + 4\Big)L_2(\widetilde{v},\widetilde{u})\\
&\quad + 2T\Big(1 + \big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\Big)L_3(\widetilde{v},\widetilde{u}) + 2TL_4(\widetilde{v},\widetilde{u}).
\end{aligned} \tag{C.80}$$

Finally, using integral Grönwall's inequality Gronwall [1919], we have:

$$\begin{aligned}
&\mathbb{E}^Z\|X(t)-Y(t)\|^2\\
&\leq 4e^{T(T+1)}L_1(\widetilde{v},\widetilde{u}) + e^{T(T+1)}\Big(2T\big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T} + 4\Big)L_2(\widetilde{v},\widetilde{u})\\
&+ 2Te^{T(T+1)}\Big(1 + \big(\|\nabla\widetilde{u}\|_\infty + \|\nabla\widetilde{v}\|_\infty\big)^2 e^{\left(\frac{1}{2}+4\|\nabla\widetilde{v}\|_\infty\right)T}\Big)L_3(\widetilde{v},\widetilde{u}) + 2Te^{T(T+1)}L_4(\widetilde{v},\widetilde{u})
\end{aligned} \tag{C.81}$$

□

C.7 Applications

C.7.1 Bounded equation domain

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 $\alpha \to 0_+$.

C.7.2 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 C.4.1.

C.7.3 Singular potential

We should augment the potential to apply our method for simulations of the atomic nucleus with Bohr-Oppenheimer approximation [Woolley and 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 $\alpha \to 0$.

C.7.4 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, 1966a]).

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 and 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 $\widetilde{u}, \widetilde{v}$, ψ_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 $\{(X_{N,j}, X'_{N,j})\}_{j=1}^{B}$.

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 and 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.

C.7.5 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.[4]

C.7.6 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 solved equations for u, v. We can get the phase and density by integrating Equation (C.4):

$$S(x,t) = S(x,0) + \int_0^t \Big(\frac{1}{2m}\langle \nabla, u(x,t)\rangle + \frac{1}{2\hbar}\left\|u(x,t)\right\|^2 - \frac{1}{2\hbar}\left\|v(x,t)\right\|^2 - \frac{1}{\hbar}V(x,t)\Big)\mathrm{d}t, \tag{C.82}$$

$$\rho(x,t) = \rho_0(x) \exp\Big(\int_0^t \big(-\langle \nabla, v(x,t)\rangle - \frac{2m}{\hbar}\langle u(x,t), v(x,t)\rangle\big)\Big)\mathrm{d}t \tag{C.83}$$

4. 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.

This allows us to define $\psi = \sqrt{\rho(x,t)}e^{iS(x,t)}$, which satisfies the Schrödinger equation (4.1). 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 (4.1) 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 and Smith, 1985].

C.8 On criticism of stochastic mechanics

Three major concerns arise regarding stochastic mechanics developed by Nelson [1966a], 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 C.7.4 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.

C.8.1 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 [1966a], 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 and 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 $\mathcal{M} = \mathbb{R}^d \setminus \{x_*\}$ 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 C.7.2.

C.8.2 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.

C.8.3 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 C.8.1. *Assume $X(t), F(t), G(t)$ are adapted to Wiener process $W(t)$ and satisfy:*

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

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

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

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, 1966a] 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.

C.8.4 Ground state

Unfortunately, our approach is unsuited for the ground state estimation or any other stationary state. FermiNet [Pfau et al., 2020a] 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.

C.9 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 and 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 (4.1) used in condensed matter physics [Serkin and 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. [2020a]. 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.

APPENDIX D
ADDITIONAL TOPICS ON F-DSM

D.1 Architecture and training details

Our F-DSM model employs a specialized neural network architecture designed to respect the antisymmetry properties of fermionic systems. In our experiments, the network consists of 8 mixing layers, 2 attention layers (each with 1 head), the embedding size is 8, number of hidden layers is 16. For the initial wave function embedding, we utilize 8 determinants. In the Jastrow factor (Equation (5.45)), we set $\alpha_{\text{anti}} = \alpha_{\text{par}} = 1.0$. Our model is implemented in JAX with double precision (float64) enabled to ensure numerical stability. As for training, we initially pretrain the model for 1000 epochs with a batch size of 8192 and learning rate of 0.01. This phase helps to establish stable initial representations before our regular training. Following pretraining, we train the model for 7000 epochs with a batch size of 32 and a learning rate of 0.001. For both phases, we employ the Adam optimizer. To ensure training stability, we use gradient clipping at 1.0. Our loss function combines three components with weights 1.0. For time integration, we split the time interval $[0, 20]$ in 1024 time steps, subsampling 256 points during training. During test sampling, we use the full 1024 time steps. Training is conducted on two NVIDIA H200 GPUs, with a total training time of approximately 10 hours. The MCMC sampling uses an initial step size of 0.5, 5000 burn-in steps, and 2000 intermediate steps, and an acceptance rate of 0.5.

For the eigenstate expansion baseline, we utilize the FullCI method from the `PySCF` library [Sun et al., 2018] with the cc-pVQZ basis set (70 basis functions). The baseline MCMC sampling employs identical parameters to our F-DSM approach: initial step size of 0.5, 5000 burn-in steps, 2000 intermediate steps, and a target acceptance rate of 0.5. Our H_2 molecule has the following parameters: nucleus coordinates $R = ((-0.7, 0.0, 0.0), (0.7, 0.0, 0.0))$, spins $S = (1, -1)$, and atomic numbers $Z = (1, 1)$.

For the computational benchmarks reported in Section 5.5, we sample $2^{16} = 65,536$ samples per time step for both methods. We utilize the same MCMC sampling parameters for the baseline as we mentioned previously. It is worth noting that our MCMC sampling implementation leverages GPU acceleration – an engineering contribution, as traditional MCMC sampling typically runs on CPUs and implementing efficient GPU-accelerated MCMC requires significant optimization. Performance measurements are collected on identical NVIDIA H200 hardware to ensure fair comparison. Without this GPU acceleration for MCMC sampling, the computation time would be substantially longer than our reported results. We believe this comparison provides a fair assessment of our approach and the baseline method.

D.2 Initialization challenges

To illustrate the difficulty of the problem and conventional quantum chemistry methods robustness, we analyze initial conditions obtained with different methods.

Figure D.1 compares ground state energy predictions obtained from HF and FullCI methods using different basis sets. As expected, FullCI consistently yields lower energies due to its more complete treatment of electron correlation. Within each method, increasing the basis set quality (from 6-31G to cc-pVDZ to cc-pVQZ) leads to systematically improved ground state energies. These variations highlight a key difficulty in quantum dynamics: the accuracy of the initial state preparation is highly sensitive to both the computational method and the numerical basis used.

Figure D.2 further compares ground and excited state predictions using FullCI across the same basis sets. As the basis set becomes more complete, both E_0 and E_1 decrease, and the energy gap shifts slightly. This illustrates that even within high-accuracy methods, the representational limitations of finite basis sets can lead to discrepancies in state energies, posing a fundamental challenge when initializing quantum dynamics from excited configurations.

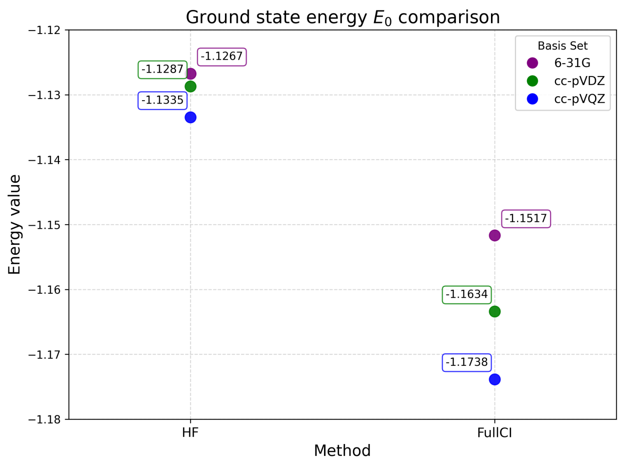


Figure D.1: Comparison of ground state energy predictions using HF and FullCI methods across various basis sets. The plot demonstrates the substantial method-dependent variation in ground state energy estimations, with FullCI consistently yielding lower energies than HF due to its comprehensive treatment of electron correlation. The improvement in energy estimation with increasing basis set quality (6-31G $\rightarrow$ cc-pVDZ $\rightarrow$ cc-pVQZ) is evident within each approach. These variations in initial condition energy estimations highlight a fundamental challenge in quantum dynamics simulations.

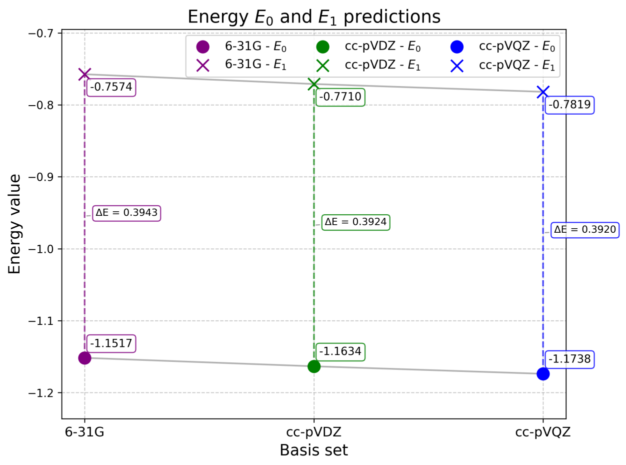


Figure D.2: Comparison of energy states predicted using different basis sets in FullCI. Circles correspond to the ground state energy E_0, and crosses indicate the excited state energy E_1. Vertical dashed lines connect the paired energy states for each basis set, with the energy difference (ΔE) explicitly labeled. The figure shows the systematic lowering of both energy states and subtle changes in energy gaps as the basis set complexity increases (6-31G $\rightarrow$ cc-pVDZ $\rightarrow$ cc-pVQZ).