UNIVERSITY OF CALIFORNIA, MERCED

Data-Driven Simulation and Inference for Time-Evolving Systems with Applications to Quantum and Electron Dynamics

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Applied Mathematics

by

Hardeep Bassi

Committee in charge:

Changho Kim, Chair

Chrysoula Tsogka

Andy Wan

Yuanran Zhu

2026

The dissertation of Hardeep Bassi is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

(Chrysoula Tsogka)

(Andy Wan)

(Yuanran Zhu)

(Changho Kim, Chair)

University of California, Merced

2026

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

This dissertation was complete with the support of many individuals detailed below.

I would first like to thank my advisor Professor Changho Kim for his guidance and unwavering support for me and my work. His honest advice and feedback strengthened my abilities in a variety of academic and personal aspects including, but not limited to, writing, speaking, and thinking. In a similar light, I would like to thank my committee members from UC Merced of Professor Chrysoula Tsogka and Professor Andy Wan. I particularly would like to thank Chrysoula for her advice/guidance, and for being an excellent and dedicated graduate chair for our department. I also would like to particularly thank Andy for asking me to discuss my research with him when he first joined our department; taking the time to stand at the whiteboard in his office and talk through what I had been working on and what I was thinking about at the time was deeply encouraging during a challenging period. These gestures left a profound impact on me and my PhD studies. Changho, Chrysoula, and Andy were unfailingly supportive of me throughout my PhD studies. Over the past four and a half years they have shaped how I approach research, from how I ask questions to how I structure and solve problems. Most importantly, they helped me learn how to think as an applied mathematician, and their insights will continue to have a lasting influence in my work.

Approximately 2 years of my PhD were spent as a visiting student researcher at Lawrence Berkeley National Laboratory (LBNL). The times I had at LBNL were undoubtedly the most enjoyable and productive times of my PhD, and in fact, all works in this dissertation were partially or fully funded by LBNL. I largely owe this positive experience to my LBNL advisors Dr. Chao Yang and Dr. Yuanran Zhu. I would like to thank Chao for providing me with funding for many semesters and letting me work on problems that I found to be interesting and inviting me to so many places to speak about our work. His insight, intuition, and ideas are unparalleled and I will forever be amazed by the breadth of knowledge that he has across a variety of academic fields. It was an absolute privilege to be able to work with him for so long and I will miss our weekly meetings and lunch outings dearly. Above all, I am especially grateful to Yuanran for bringing me into LBNL initially and for mentoring me throughout my stay here. Though he (and probably anyone that I've talked to regarding my experience at LBNL) probably already knows this, his academic and personal support during my PhD studies was truly unmatched, and he served and will continue to serve as a constant source of inspiration for me. I'll always fondly look back on our conversations regarding research and life. Though he doesn't need it, I wish him great success in his academic career and am beyond fortunate to have worked alongside him for most of my research.

The community at LBNL, and in particular the Scalable Solvers Group, was among

the most welcoming, knowledgeable, and enjoyable environments a graduate student in applied mathematics could experience. First, I thank Dr. Roel Van Beeumen and Dr. Yizhi Shen both very much for accepting me onto and allowing me to formulate the FDODMD project, which allowed me to learn about quantum computing and its applications to quantum chemistry. To Roel, thank you for explaining and showing to me that quantum computing is largely viewable through the lens of linear algebra and for always inviting me to group outings and checking in. To Yizhi, thank you for diligently reviewing all of the work we wrote together and for your immense patience as I learned the fundamentals of a new field. I also thank Dr. Erika Ye, as her talk at SIAM NCC24 led to the initial formulation of the SRaFTE project. She repeatedly went out of her way to help, contribute, and advise within this project with remarkable kindness. I am deeply grateful for her support and feel privileged to have worked with a world-class researcher such as herself. Similarly, I thank Dr. Pu Ren for his support and feedback on the SRaFTE project. I had first reached out to gain his opinion on the project when I was in need of an external technical perspective. His careful evaluation and encouragement helped me recognize the value in the SRaFTE project and was a big source of motivation to continue working on it. Over time, that initial conversation grew into a wonderful collaboration, and I am thankful for his contributions and perspectives throughout the past year. Finally, I thank all other members of the Scalable Solvers Group that I was fortunate enough to interact with on a regular basis, including Professor Senwei Liang, Dr. Alec Dektor, Dr. Jia Yin, Dr. Dong Min Roh, Dr. Pinchen Xie, Dr. Xiaoye Li, and Lucy Radcliffe, for fostering such a supportive and collegial environment and for the many conversations, guidance, and help along the way. Everyone in this group taught me by example how to formulate and ask the correct questions to go about solving complex problems in the computational sciences.

To Professor Hanbaek Lyu and Richard Yim, thank you both for giving me such a strong and encouraging first experience in research. Working with you both was the beginning of my adventure in academia, and it inspired me to continue down the path of research with both curiosity and confidence.

There were also many members of the UC Merced Applied Mathematics community that were instrumental to any success that I had. Thank you to Professor Shilpa Khatri and Professor Suzanne Fernandes-Sindi for the opportunity to TA your courses and for running them with such clarity and care. Some of my most enjoyable times on campus came from TA-ing the discussion sections from your classes, and I greatly appreciated the chance to engage with both of you and with your students. I also thank my 2021 cohort in the Applied Mathematics department for being so close-knit and for supporting one another throughout this journey; the sense of community we built made a real difference for me. To this end, I am grateful to Matthew Blomquist, Zihan Xu, Yu Lu, Irabiel

Romero, Mohammed Aburidi, and Morgan Lavestein-Bendall, and especially to Matthew, Zihan, Yu, and Irabiel for the many conversations and perspectives they shared with me. Similarly, thank you to those in the cohorts before mine for creating such a welcoming and warm community during my early years. This includes Dr. A. Ali Hyedari, Dr. Alex Ho, Dr. Majerle Reeves, and Dr. Jocelyn Ornelas-Munoz. I must also thank Professor Christine M. Isborn for providing me with patient guidance during my early years and explaining quantum chemistry topics in ways that catered to my understanding, and for financial support during my second year. Finally, I thank Professor Harish S. Bhat for advising me on many projects and for financial support throughout my PhD.

In the interest of space, I acknowledge the friends that I have outside of my own research who supported me throughout my PhD. Their steady presence carried me through the most demanding parts of this journey. These include, but are not limited to: Zachary Brown, Dylan Buer, Emily Chang, Robin Chang, Abhishek Devarajan, Halie Gonzalez, Hannah Ho, Shawn Hosain, Daniel Jiang, Daniel Jiang-Nussbaum, Yunmeng Jiang, Kunal Kathuria, Alexander Khazatsky, Victoria Le, Gary Liu, Eric Mathew, Kaishu Miyao, Caitlin Moore, Nicholas Moore, Benjamin Pearce, Tyler Stejskal, Isaac Tang, Oreste Turchetti, Edmund Wang, Benjamin Wu, Daniel Wu, Jessica Xu, and Christopher Yeh.

Finally, I thank my family members for their support: Ranvir Bassi, Balraj Singh, Jax Bassi, Pavinpreet Sanghera, Harendeep Sanghera, Sienna Sanghera, Ranjit Dosanjh, Simran Dosanjh, Devin Dosanjh, Gurpreet Cheema, and Gavin Cheema.

Publications. Portions of this dissertation are based on previously published work. Chapter 2 is based on H. Bassi et al., *Learning nonlinear integral operators via recurrent neural networks and its application in solving integro-differential equation*s, in Machine Learning with Applications (2024), and is reused under a Creative Commons Attribution (CC BY 4.0) license. Chapter 4 is based on H. Bassi et al., *From noisy observables to accurate ground-state energies: A quantum–classical signal subspace approach with denoising*, in APL Quantum (2025), and is reused under a Creative Commons Attribution-Non Commercial (CC BY-NC 4.0) license.

Hardeep Bassi

hbassi2@ucmerced.edu || https://scholar.google.com/citations?user=wJqFPgwAAAAJ || (209) 324-9580

Research interests

AI for science

Theoretical and applied scientific machine learning

Interests: Scientific computing, multiscale modeling, operator learning, physics-constrained neural network design, super-resolution, neural spatiotemporal dynamic modeling, neural network weight-space analysis, transfer learning and generalization of foundation models, physics of language models, reduced-order modeling

Physical sciences

Quantum chemistry, quantum computing, quantum physics

Interests: Ground state energy estimation, density functional theory, quantum computing for quantum chemistry applications, phase transition, quantum many-body theory, condensed-matter physics

Education

University of California, Merced Graduation: 01/26
PhD Applied Mathematics

University of California, Berkeley Graduation: 05/21
B.A. Applied Mathematics with Computer Science concentration

Technical Skills

Languages: Python, MATLAB, SQL, Java
Tool/Technologies: AWS, Git, Docker, Google Cloud Platform, Slurm, LaTeX
Libraries: JAX, PyTorch, Tensorflow, sci-kit learn, cvxpy, pandas, numpy, matplotlib, cupy

Publications and in progress manuscripts

SRaFTE: Super-Resolution and Future Time Extrapolation for Time-Dependent PDEs
Submitted to *APL Computational Physics*
Hardeep Bassi, Yuanran Zhu, Erika Ye, Pu Ren, Alec Dektor, Harish S. Bhat, and Chao Yang
arXiv: https://arxiv.org/abs/2509.12220

Sigma-Attention: A Transformer-based operator learning framework for self-energy in strongly correlated systems
Submitted to *Physical Review B*
Yuanran Zhu, Peter Rosenberg, Zhen Huang, **Hardeep Bassi**, Chao Yang, Shiwei Zhang
arXiv: https://arxiv.org/pdf/2504.14483

From noisy observables to accurate ground state energies: a quantum classical signal subspace approach with denoising
Published in *APL Quantum*
Hardeep Bassi, Yizhi Shen, Harish S. Bhat, and Roel Van Beeumen
arXiv: https://arxiv.org/pdf/2505.10735

Incorporating memory into propagation of 1-electron reduced density matrices
Published in *Journal of Mathematical Physics*
Harish S. Bhat, **Hardeep Bassi**, Karnamohit Ranka, Christine M. Isborn
arXiv: https://arxiv.org/pdf/2403.15596

Learning nonlinear integral operators via recurrent neural networks and its application in solving integro-differential equations
Published in *Machine Learning with Applications*
Hardeep Bassi, Yuanran Zhu, Senwei Liang, Jia Yin, Cian C. Reeves, Vojtěch Vlček, and Chao Yang
arXiv: https://arxiv.org/abs/2310.09434

Learning to predict the synchronization of coupled oscillators on randomly generated graphs
Hardeep Bassi, Richard Yim, Rohith Kodukula, Joshua Vendrow, Cherlin Zhu, Hanbaek Lyu
Published in *Scientific Reports*
arXiv: https://arxiv.org/abs/2012.14048

Work Experience

Protegrity November 2025 – Present
Lead Machine Learning Engineer
Lead machine learning engineer for the Generative AI division. Responsibilities and active areas of research include:

- Implementing an agentic multi-hop retrieval pipeline that parses hierarchical internal documentation in response to customer queries and generates instructions for correct and secure Protegrity product use aligned with customer policies.
- Designing synthetic data generation methods with rigorous privacy guarantees, enabling shareable datasets that preserve statistical utility without exposing sensitive records.

Research

Lawrence Berkeley National Laboratory June 2023 – Present
Graduate Student Researcher *June 2023 – January 2026*
Research Affiliate *February 2026 – Present*
Graduate student researcher advised by Dr. Chao Yang and Dr. Yuanran Zhu. Research includes:

- Designed operator learning framework with RNNs to accelerate solvers for nonlinear integro-differential equations in solid-state many-body systems.
- Assisted in designing a transformer-based operator learning framework for learning the electronic self-energy in strongly correlated fermionic systems.
- Created a noise robust ground-state energy estimation hybrid quantum-classical algorithm via data-driven spectral estimation on noisy quantum signals obtained from near-term quantum hardware.
- Developed SRaFTE, a two-phase super-resolution and spatiotemporal forecasting method.

University of California, Merced March 2022 – January 2026
Graduate Student Researcher
Graduate student researcher advised by Professor Changho Kim. Research includes:

- Quantified the effect of memory in time-delayed equations of motion for reduced 1-electron density propagation.
- Learned electron-exchange correlation potentials from time series data of electron densities.

University of Wisconsin - Madison June 2020 – May 2022
Lead Undergraduate Researcher
Team research lead advised by Professor Hanbaek Lyu. Research included:

- Developed synchronization classification models in coupled oscillator systems.

Selected Invited Talks

LBL–ICSI Joint Seminar for AI and Statistics, *Berkeley, CA* — January 2026
C2SEPEM Seminar, *Berkeley, CA* — December 2025
SIAM NCC25, *Berkeley, CA* — October 2025
SIAM CSE25, *Fort Worth, TX* — March 2025
SIAM NCC24, *Merced, CA* — October 2024
UCM Scientific Computing and Data Science Seminar, *Merced, CA* — October 2024
SIAM MDS24, *Atlanta, GA* — October 2024
UCM Scientific Computing and Data Science Seminar, *Merced, CA* — March 2024
UCM Scientific Computing and Data Science Seminar, *Merced, CA* — March 2023
UCM Scientific Computing and Data Science Seminar, *Merced, CA* — April 2022

Mentoring & Supervision

Lawerence Berkeley National Laboratory — May 2025 – August 2025
Undergraduate Research Mentor

- Mentored 2 undergraduate researchers from Rensselaer Polytechnic Institute (RPI) on implementing FDODMD and other ground-state energy estimation hybrid algorithms on quantum hardware at RPI (IBM Quantum System One).

Awards

UC Merced Fletcher Jones Fellowship (declined) — July 2025
SIAM CSE25 Travel Fellowship — January 2025
UC Merced Applied Mathematics Graduate Travel Fellowship — October 2024
UC Merced MacKenzie Scott Graduate Student Supplemental Travel Award — October 2024
UC Merced Applied Mathematics Excellence in Teaching Award — May 2024
UC Merced NSF RTG Fellowship — August 2022 – May 2023

Service

Journal reviewer
Journal of Computational Physics

Teaching

University of California, Merced
Teaching Assistant

- **Math 24**: Linear Algebra and Differential Equations (Spring 2024) — **Rating: 6.7/7 and 6.7/7**
- **Math 24**: Linear Algebra and Differential Equations (Fall 2023) — **Rating: 6.9/7 and 6.7/7**
- **Math 141**: Linear Analysis (Spring 2022) — **Rating: 6.8/7 and 6.9/7**
- **Math 24**: Linear Algebra and Differential Equations (Fall 2021) — **Rating: 6.6/7 and 6.8/7**

University of California, Berkeley
Course Staff

- **CS61B**: Data Structures (Spring 2020)
- **CS70**: Discrete Mathematics and Probability (Summer 2019)

ABSTRACT OF THE DISSERTATION

Data-Driven Simulation and Inference for Time-Evolving Systems with Applications to Quantum and Electron Dynamics

by Hardeep Bassi
Doctor of Philosophy in Applied Mathematics

University of California Merced, 2026

Committee Chair: Changho Kim

Time-evolving systems arise throughout science and engineering whenever a governing equation is used to describe how a state evolves in time. A central goal in these settings is to accurately simulate, forecast, or infer this evolution, but practical progress is often limited by a combination of computational and data constraints. Common obstacles include expensive multiscale discretizations, temporal nonlocality induced by memory terms, and limited observations available for calibration. These difficulties are especially acute in quantum and electron dynamics, where relevant state spaces grow exponentially and near-term quantum hardware yields imperfect, noisy measurements. To this end, this dissertation develops equation-aware, data-driven methods that bring expensive, often intractable, computations within practical reach while preserving accuracy on targeted objectives. Specifically, the contributions are threefold.

(1) For integro-differential equations (IDEs) motivated by the Kadanoff–Baym equations, explicit history integrals force numerical solvers to scale quadratically in the number of time steps. Using an analytic result from many-body perturbation theory that implies a deterministic map from the instantaneous Green's function to the history integral, we replace the history integral with a time-local update learned by a neural network and obtain linear-in-time cost at target accuracy.

(2) For spatiotemporal systems with only short paired coarse–fine observation windows, we propose a coarse-to-fine super-resolution and forecasting framework that couples a low-cost coarse integrator with a learned super-resolution operator to emulate fine-grid evolution without fine-grid time stepping. We evaluate the approach on canonical partial differential equations (2D heat, wave, and incompressible Navier–Stokes) and a reduced Vlasov–Poisson system for electron motion in a plasma, quantify accuracy versus compute/storage, and provide a novel error analysis for long-horizon error accumulation.

(3) For ground-state energy estimation of molecular Hamiltonians from noisy time-evolution signals acquired on near-term quantum hardware, we introduce a noise-robust post-processing algorithm that achieves chemical accuracy with reduced quantum overhead and reduced sensitivity to hyperparameters.

Taken together, **(1)**–**(3)** show that targeting computational bottlenecks in systems with modular data-driven components, guided by the equations of motion and the measurement process, can substantially improve efficiency and robustness, whilst also maintaining accuracy on the targeted simulation and inference objectives.

Chapter 1

Introduction

Time-evolving systems are ubiquitous across nearly every domain of science and engineering, arising whenever one seeks to describe how a state evolves under a governing equation of motion. Examples range across a variety of fields and disciplines including fluid and plasma dynamics, climate models, and electron dynamics in materials and molecules [14, 197, 198, 30]. The practical barriers to accurate simulation of these systems, however, are rarely tied to a single source. They instead arise from a variety of computational and data limitations. In many settings, resolving the relevant spatial structure of a simulation of interest pushes the state space representation into high and often times intractable dimensions. In other settings, the primary computational burden is temporal in nature, where key behavior becomes visible only over long horizons or through interactions across disparate time scales. In turn, this can induce stiffness and demand carefully tuned, expensive integration procedures. Finally, when the goal is to incorporate information from experimental data, additional limitations arise because measurements are often noisy or low precision. Consequently, this further reduces data fidelity and amplifies uncertainty in downstream modeling and prediction.

The previously mentioned computational roadblocks are particularly acute in quantum and electron dynamics, which form the main application suite of this dissertation. These dynamics underlie critical physical phenomena such as charge transport, light–matter interaction, and superconductivity in materials. As a result, reliable simulation at realistic scales remains a central goal across fields such as quantum chemistry and condensed-matter physics. Meeting this goal in practice is not only a matter of better physical modeling, but also of algorithmic and implementation choices. Meaningful reductions in computational cost therefore sit at the intersection of applied mathematics, physics, and computer science and require tremendous synergy between this trinity. Consequently, numerous computational methods and formalisms have been

developed in recent decades by computational chemists and physicists to simulate these particular dynamical systems.

Ab-initio electronic-structure methods aim to predict electronic properties directly from the underlying quantum mechanics, with minimal reliance on empirical fitting, making them broadly useful across molecules and materials. Time-dependent density functional theory (TDDFT) [120, 100] is a widely used ab-initio approach for nonequilibrium dynamics that propagates effective single-particle (Kohn–Sham) orbitals in time, often in large basis sets, and consequently exhibits steep scaling with system size. Another formalism, based on non-equilibrium Green's functions (NEGF), describes two-time correlators that satisfy sets of integro-differential equations with non-Markovian memory kernels [201, 142, 168]. In condensed-matter and nonequilibrium quantum many-body physics, this leads to the Kadanoff–Bayms equations, where memory integrals over the past history of the Green's function induce quadratic or worse scaling in the number of time steps. Both TDDFT and the NEGF formalism share striking mathematical similarities, despite being derived from different theoretical frameworks: in both cases, the equations of motion contain an analytically unknown term that encodes electron–electron exchange and correlation and, when relevant, coupling to an external environment (the exchange–correlation potential in TDDFT and the self-energy in NEGF), and the two approaches trade off different aspects of the curse of dimensionality and temporal nonlocality. Critically, both formalisms also become prohibitively computationally expensive when pushed toward realistic system sizes, strong correlation and driving, or long-time behavior [52, 89, 131, 130, 167, 57, 205, 206, 29, 28, 100, 212].

Computational issues also manifest in new ways within the realm of quantum computing, where key quantities such as ground-state energies and other spectral properties of molecular Hamiltonians can be inferred from the frequency content of experimentally accessible time-evolved signals. Such examples include overlap or correlation measurements obtained by implementing unitary time evolution under the Hamiltonian on the quantum device. Quantum hardware can, in principle, represent and simulate molecular Hamiltonians with resources that scale polynomially in system size, thereby avoiding the exponential cost of exact classical methods. This is particularly compelling in quantum chemistry because, within the Born–Oppenheimer approximation, the electronic ground-state energy together with the nuclear–nuclear repulsion defines the potential energy surface, which in turn governs equilibrium structures, forces, and reaction pathways without reliance on empirically fitted interatomic potentials. In the near term, however, the main obstacle is quantum hardware noise, since the time-domain signals used to infer energies are corrupted by decoherence, gate errors, and finite-shot noise [27, 106].

Alongside these problems in the realm of quantum chemistry and computing,

classical electron dynamics in plasmas, governed by the Vlasov–Poisson system, provide a complementary testbed where spatial resolution, long-time accuracy, and compute/storage tradeoffs push the limits of classic numerical solvers [197, 198]. Kinetic plasma models underpin applications ranging from fusion devices to space and astrophysical plasmas [9, 55]. At the same time, even in reduced settings, the evolved electron distribution function develops fine filamentary structure and strong phase-space mixing, hence faithfully capturing kinetic effects of physical interest requires high resolution in both position and velocity over long time horizons, with correspondingly large memory and runtime costs.

Across all of the previously mentioned settings, the methodological questions of how to identify and learn the key quantities of interest, and how to integrate them with existing solvers or hardware, play out in particularly rich and challenging ways. Holistically, these challenges have motivated a broad family of data-driven approaches that aim to develop surrogate or reduced order models for the original evolution problem. Classical reduced-order modeling (ROM) techniques such as reduced-basis methods, proper orthogonal decomposition (POD), and Krylov subspace methods exploit the fact that many solutions live on low-dimensional manifolds embedded in a high-dimensional state space [22, 18, 179]. By projecting the dynamics onto a carefully selected reduced basis, one obtains a lower-dimensional system that is faster to solve while retaining the key characteristics of the full, high-dimensional system. Complementary tools, including dynamic mode decomposition (DMD) and related Koopman approximations, extract coherent spatiotemporal modes from time-series data and can approximate time evolution in a suitable function space [18, 201, 143, 116, 129].

More recently, advances in machine learning (ML) have also led to powerful surrogate models for time-dependent simulations. Applications span from using recurrent neural networks and sequence-based models to emulate time integrators, transformer architectures to learn spatiotemporal dynamics, and auto-encoder based ROMs to learn low-dimensional latent dynamics [17, 84, 48, 63, 193, 176, 45]. Another branch of ML-based surrogates uses operator learning, in which neural networks approximate mappings between function spaces rather than finite-dimensional vector spaces [90]. Here, neural operators such as DeepONets and Fourier neural operators aim to learn solution maps that can be evaluated at arbitrary spatial and temporal locations and interpolated across families of input functions [102, 111, 112, 110].

Within both the classic ROM and ML literature, one can distinguish between two broad strategies. On one hand, equation-free or propagator-learning methods attempt to learn the full evolution map $u(t) \mapsto u(t + \Delta t)$ direct from data, with minimal use of the governing equations beyond the data generation process [111, 102]. On the other hand, PDE-constrained methods use the equations of motion to inform what to learn and how to

parametrize unknown closures [56, 47, 18, 12]. In the latter, a common approach is to derive a reduced equation of motion with an unclosed term (such as a memory kernel or effective potential), and train a neural network to approximate this term, via backpropagation through the underlying equation of motion during training. A complementary approach, which this dissertation adopts, uses the equations of motion not as hard constraints during training, but rather, as *structural guidance*. Here, data-driven models are applied in conjunction with standard workflows and do not strictly enforce the governing equations as hard constraints during training. The methods developed in this dissertation (i) replace the dominant memory term in an equation of motion with a learned neural operator, (ii) pair an inexpensive coarse-grid propagator with a neural super-resolution map to form an effective fine-grid propagator at a cost proportional to the coarse-grid propagator, and (iii) use a dimensionality-reduction scheme to robustly extract spectral information from noisy time-evolution signals. This occupies a middle ground between equation-free methods and methods that directly enforce the equations of motion, which we refer to here as an *equation-aware* approach.

Concretely, this dissertation is organized as follows around three such case studies. In Chapter 2, we develop an RNN-based surrogate that replaces the quadratic-cost history integral in a class of IDEs inspired by the Kadanoff–Baym equations with a learned time-local map, yielding linear-time propagation at target accuracy. In Chapter 3, we introduce a coarse to fine grid super-resolution and forecasting framework (SRaFTE) that couples a low-cost coarse-grid time integrator with a learned spatial super-resolution operator to avoid fine-grid time stepping while still recovering fine-scale dynamics, and its application to plasma dynamics. Finally, in Chapter 4, we present a noise-robust ground-state energy estimation algorithm using the machinery of DMD on noisy time-evolution signals produced by noisy, near-term quantum hardware, thereby reducing quantum computational overhead while maintaining chemical accuracy. A concluding chapter synthesizes these results, highlights common design principles across the three settings, and outlines open directions for data-driven modeling and inference in time-dependent systems and its applications to electron and quantum dynamics.

The unifying theme in all of these works is to recast each problem so that the most onerous aspect of the computation—be it nonlocal history integrals, fine-grid propagation, or noisy observations—is handled by a tailored data-driven operator or estimator, that is tightly integrated with the underlying equations of motion. Taken together, the results of this dissertation demonstrate that data-driven modeling can alleviate canonical computational bottlenecks in time-evolving systems and support accurate simulation and dependable inference of key dynamics and observables in quantum and electron dynamics.

1.1 Primer on quantum mechanics

To provide insight, we briefly review the main quantum dynamics formalisms that motivate the first and third case studies. We start from the many-electron Schrödinger equation and its second-quantized representation, then summarize density functional theory (DFT) and its time-dependent extension (TDDFT), and finally describe the non-equilibrium Green's function (NEGF) formalism.

At the most fundamental level, an interacting N-electron system is described by the time-dependent Schrödinger equation (TDSE)

$$i\partial_t \Psi(\mathbf{r}_1,\dots,\mathbf{r}_N,s_1,\dots,s_N;t) = \hat{H}\,\Psi(\mathbf{r}_1,\dots,\mathbf{r}_N,s_1,\dots,s_N;t), \tag{1.1}$$

where Ψ is the antisymmetric N-body wavefunction on a $3N$-dimensional configuration space, $\mathbf{r}_i \in \mathbb{R}^3$ and s_i denote the position and spin of the ith electron, and $\hat{H}$ is the molecular Hamiltonian. In first quantization, a typical electronic Hamiltonian with a fixed nuclei assumption under the Born-Oppenheimer approximation has the form

$$\hat{H} = \sum_{i=1}^{N}\Big(-\frac{1}{2}\nabla_i^2 + v_{\text{ext}}(\mathbf{r}_i)\Big) + \sum_{1\le i<j\le N}\frac{1}{|\mathbf{r}_i-\mathbf{r}_j|}, \tag{1.2}$$

where the one-body operator $-\frac{1}{2}\nabla_i^2$ represents electronic kinetic energy, v_{ext} is the external potential induced by fixed nuclei and any applied fields, with the final term being the two-body Coulomb term that accounts for electron–electron repulsion.

A convenient representation for quantum chemistry calculations is obtained by expanding the electronic degrees of freedom in a finite one-particle basis $\{\varphi_p(\mathbf{r},s)\}_{p=1}^{M}$ (e.g., atomic or molecular orbitals) and passing to second quantization. We introduce fermionic creation and annihilation operators $a_p^\dagger$ and a_p, which create or destroy an electron in orbital φ_p, and satisfy the canonical anticommutation relations

$$\{a_p, a_q^\dagger\} = \delta_{pq}, \qquad \{a_p, a_q\} = \{a_p^\dagger, a_q^\dagger\} = 0.$$

In this basis, the Hamiltonian can be written as

$$\hat{H} = \sum_{p,q=1}^{M} h_{pq}\, a_p^\dagger a_q + \frac{1}{2}\sum_{p,q,r,s=1}^{M} v_{pqrs}\, a_p^\dagger a_q^\dagger a_s a_r, \tag{1.3}$$

where

$$h_{pq} = \int \varphi_p^*(\mathbf{r},s)\Big(-\tfrac{1}{2}\nabla^2 + v_{\text{ext}}(\mathbf{r})\Big)\varphi_q(\mathbf{r},s)\,d\mathbf{r}\,ds, \tag{1.4}$$

$$v_{pqrs} = \iint \frac{\varphi_p^*(\mathbf{r},s)\,\varphi_q^*(\mathbf{r}',s')\,\varphi_r(\mathbf{r},s)\,\varphi_s(\mathbf{r}',s')}{|\mathbf{r}-\mathbf{r}'|}\,d\mathbf{r}\,d\mathbf{r}'\,ds\,ds' \tag{1.5}$$

are the one- and two-electron integrals, respectively. The first term in (1.3) is a bilinear form in $(a_p^\dagger, a_q)$ capturing kinetic and external potential energies, while the second is a quartic form encoding pairwise Coulomb interactions. The Hilbert space dimension grows combinatorially with N and M (roughly $\binom{M}{N}$ for fixed particle number), so direct wavefunction-based solution of the Schrödinger equation is intractable even for moderately sized systems [120].

Density functional theory [120] addresses the curse of dimensionality by recasting the ground-state problem in terms of the one-body electron density

$$n(\mathbf{r}) = N \sum_{s_1,\dots,s_N} \int_{\mathbb{R}^{3(N-1)}} |\Psi(\mathbf{r}, \mathbf{r}_2, \dots, \mathbf{r}_N, s_1, \dots, s_N)|^2 \, d\mathbf{r}_2 \cdots d\mathbf{r}_N, \tag{1.6}$$

rather than the full N-body wavefunction. Standard theoretical results guarantees that the ground-state energy can be written as a functional

$$E[n] = T_s[n] + \int v_{\text{ext}}(\mathbf{r}) n(\mathbf{r}) \, d\mathbf{r} + E_{\text{H}}[n] + E_{\text{xc}}[n], \tag{1.7}$$

where $T_s[n]$ is the kinetic energy of a fictitious non-interacting system with density n, $E_{\text{H}}[n]$ is the classical Hartree electrostatic energy

$$E_{\text{H}}[n] = \frac{1}{2} \iint \frac{n(\mathbf{r}) n(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} \, d\mathbf{r} \, d\mathbf{r}',$$

and $E_{\text{xc}}[n]$ is the analytically unknown exchange–correlation functional, which collects all many-body effects beyond Hartree theory and must be approximated in practice.

In Kohn–Sham DFT, one introduces single-particle orbitals $\{\varphi_i\}_{i=1}^{N_{\text{occ}}}$ whose density recovers $n(\mathbf{r}) = \sum_i |\varphi_i(\mathbf{r})|^2$ and solves the self-consistent eigenvalue problem

$$\left(-\frac{1}{2}\nabla^2 + v_{\text{ext}}(\mathbf{r}) + v_{\text{H}}[n](\mathbf{r}) + v_{\text{xc}}[n](\mathbf{r}) \right) \varphi_i(\mathbf{r}) = \varepsilon_i \varphi_i(\mathbf{r}), \tag{1.8}$$

where N_{occ} denotes the number of occupied orbitals, $v_{\text{H}}[n] = \delta E_{\text{H}}/\delta n$, and $v_{\text{xc}}[n] = \delta E_{\text{xc}}/\delta n$ the Hartree and exchange–correlation potentials, respectively. Mathematically, (1.8) is a nonlinear eigenproblem on a one-electron Hilbert space. Numerically, one discretizes space or an orbital basis to obtain a matrix eigenproblem whose cost typically scales as $\mathcal{O}(M^3)$ or $\mathcal{O}(M^4)$ in the number of basis functions M, due mainly to Fock-like matrix construction and diagonalization. Modern algorithms and HPC allow DFT to routinely treat systems with hundreds of atoms, but systems with thousands of atoms, complex geometries, or many parameter sweeps remain computationally demanding [98, 3, 15, 73, 192].

Time-dependent density functional theory [100, 59, 182] extends the DFT framework to excited states and real-time dynamics. In real-time TDDFT, one evolves a set of time-dependent Kohn–Sham orbitals $\varphi_i(\mathbf{r},t)$ according to

$$i\partial_t\varphi_i(\mathbf{r},t) = \Big(-\frac{1}{2}\nabla^2 + v_{\text{ext}}(\mathbf{r},t) + v_{\text{H}}[n](\mathbf{r},t) + v_{\text{xc}}[n](\mathbf{r},t)\Big)\varphi_i(\mathbf{r},t), \tag{1.9}$$

with density $n(\mathbf{r},t) = \sum_i |\varphi_i(\mathbf{r},t)|^2$. The Hartree potential is given by an instantaneous Poisson-type integral of $n(\mathbf{r},t)$, while the time-dependent exchange–correlation potential $v_{\text{xc}}[n](\mathbf{r},t)$ is usually approximated by adiabatic functionals that depend on $n(\cdot,t)$ at the current time. In principle, a memory effect is known to analytically exist, however only recently has work been developed to quantify this effect [11]. Errors arising from the adiabatic approximation of the exchange-correlation potential is quantified by a plethora of previous work [59, 93, 114, 141].

In practice, (1.9) is discretized in space and integrated in time with small time steps to resolve electronic timescales, leading to the solution of a large coupled system of linear or nonlinear ODEs at each step. The cost of evaluating v_{xc}, constructing effective Hamiltonians, and applying them to all orbitals yields overall scaling that can be quartic in system size in the worst case. Even with GPU-accelerated implementations and careful numerics, long-time first-principles simulations of electron dynamics in large molecules or materials remain at the edge of feasibility [125].

For systems driven far from equilibrium, such as electron gases under strong laser fields or strongly correlated systems, mean-field orbital dynamics can be insufficient [60, 135, 164, 145]. The non-equilibrium Green's function formalism provides a powerful alternative framed in terms of two-time correlation functions rather than wavefunctions [168]. In second quantization, one introduces the field operator

$$\hat{\psi}(\mathbf{r},t) = \sum_p \varphi_p(\mathbf{r})\, a_p(t),$$

and defines the time-ordered Green's function

$$G(1,2) = -i\,\langle \mathcal{T}_C\, \hat{\psi}(1)\, \hat{\psi}^\dagger(2)\rangle, \qquad 1 = (\mathbf{r}_1,t_1),\ 2 = (\mathbf{r}_2,t_2), \tag{1.10}$$

where $\mathcal{T}_C$ denotes contour-time ordering on the Keldysh contour. Other components, such as the retarded G^R and lesser $G^<$ Green's functions, are obtained from G and encode spectral and occupation information, respectively. In a finite one-particle basis, these objects become $M \times M$ matrix-valued functions of two time variables, e.g.,

$$G^<_{pq}(t,t') = i\,\langle a_q^\dagger(t') a_p(t)\rangle. \tag{1.11}$$

The evolution of G is governed by the Kadanoff–Baym equations (KBEs). In a one-particle basis and suppressing orbital indices, these can be written schematically as

$$\Big(i\partial_t - h(t)\Big)G(t,t') = \delta(t-t') + \int_{t_0}^{t} \Sigma(t,\tau)\, G(\tau,t')\, d\tau, \tag{1.12}$$

along with a corresponding adjoint equation in t'. Here, $h(t)$ is the mean-field one-body Hamiltonian, and $\Sigma(t,\tau)$ is the self-energy kernel, which summarizes the effect of all two-body interactions and many-body correlations on the single-particle dynamics. Formally, Σ is a nonlinear functional of G; specific approximations (e.g., Feynman diagrams) produce explicit expressions for $\Sigma[G]$ in terms of G and auxiliary propagators [180].

The convolution term

$$\int_{t_0}^{t} \Sigma(t,\tau)\, G(\tau,t')\, d\tau$$

acts as a memory integral: the value of G at time t depends on its entire past history through a kernel Σ that itself depends on G. If one discretizes time on a grid $\{t_n\}_{n=0}^{N_t}$, the unknowns become arrays $G_{pq}(t_n,t_m)$ with $\mathcal{O}(M^2N_t^2)$ storage, and the history integral requires nested time sums whose naive evaluation scales as $\mathcal{O}(N_t^3)$ per orbital block. For realistic simulations, N_t can reach 10^4 or higher, so the combination of quadratic storage and cubic time complexity renders straightforward KBE propagation prohibitively expensive.

In practice, the self-energy Σ is also analytically unknown and must be approximated. Diagrammatic truncation schemes such as GW replace the full self-energy by a subset of Feynman diagrams constructed from the single-particle Green's function and a screened interaction, yielding specific convolution kernels that remain history-dependent but are more tractable. An alternative route is furnished by the generalized Kadanoff–Baym ansatz (GKBA) [142, 168, 143], which reconstructs two-time Green's functions from single-time quantities, such as the one-particle density matrix, and effective propagators. At a high level, GKBA reduces the full two-time problem to a closed evolution equation for single-time objects, thereby alleviating the storage and complexity burdens associated with the full KBE, at the cost of approximating memory effects and certain two-particle correlations.

In summary, time-dependent electronic-structure methods such as TDDFT and NEGF/KBEs face a double burden of rapid growth in the number of degrees of freedom (orbitals, basis functions, or modes) and the presence of memory-dependent integrals that couple each time to its entire history. These features are mathematically expressed through large coupled systems of PDEs or IDEs with nonlocal kernels and high-dimensional operator coefficients, and they are the main source of computational

intractability in the problems considered here. The first case study in this dissertation isolates the memory-integral structure in a Dyson-type equation for the Green's function and learns an effective short-memory surrogate for the history term, while the third case study works directly with the unitary time-evolution operator $e^{-i\hat{H}t}$ of (1.3), using noisy time-evolution signals obtained from real and simulated quantum hardware to infer spectral properties such as the ground-state energy of $\hat{H}$. In both settings, the focus is on identifying and surrogating specific operators that possess the key physics of interest, yet are responsible for the worst computational scaling, and then integrating these surrogating into more efficient numerical pipelines.

For sake of clarity, we provide a table below for commonly used abbreviations throughout this dissertation.

Table 1.1: Common abbreviations and their meanings by chapter.

Abbreviation	Meaning	Where used
1D1V	One-dimensional in physical space and one-dimensional in velocity space (phase-space grid for Vlasov–Poisson).	Ch. 3 App. B
AR	Autoregressive; one-step map $\Phi_\theta : \hat{u}^n \mapsto \hat{u}^{n+1}$.	Ch. 3
DFT	*Density functional theory*; electronic-structure formalism based on the ground-state density.	Ch. 1
DFT	*Discrete Fourier transform* of the observable time series, used for Fourier-based denoising and thresholding.	Ch. 4, App. B
DMD	Dynamic mode decomposition; data-driven decomposition into temporal modes / frequencies.	Ch. 1, Ch. 4
FDODMD	Fourier denoising observable dynamic mode decomposition; Fourier denoised ODMD variant.	Ch. 4
GKBA	Generalized Kadanoff–Baym ansatz; reconstructs two-time Green's functions from single-time quantities to reduce memory cost.	Ch. 1
GSE	Ground-state energy of the many-body Hamiltonian.	Ch. 4, App. B

Continued on next page

Abbrev.	Meaning	Where used
GW	Diagrammatic self-energy truncation built from the Green's function G and a screened interaction W.	Ch. 1
IDE	Integro-differential equation(s) with nonlocal memory terms.	Ch. 1, Ch. 2
KBE	Kadanoff–Baym equation(s) governing nonequilibrium Green's functions.	Ch. 1, Ch. 2
LSTM	Long short-term memory recurrent neural network architecture.	Ch. 2
LW	Second-order Lax–Wendroff finite-difference scheme for the 1D1V Vlasov–Poisson time integrator.	App. A
ML	Machine learning, broadly referring to learned data-driven models.	Ch. 1–3
MODMD	Multi-Observable ODMD; extension of ODMD to multiple observables.	Ch. 4
NEGF	Non-equilibrium Green's function formalism for time-dependent many-body systems.	Ch. 1
NISQ	Noisy intermediate-scale quantum devices (near-term quantum hardware regime).	Ch. 1, Ch. 4
NSE	Navier–Stokes equations (2D incompressible vorticity formulation).	Ch. 3, App. A
ODE	Ordinary differential equation.	Ch. 2–3
ODMD	Observable Dynamic Mode Decomposition; DMD applied to scalar observable time series from a quantum device.	Ch. 4, App. B
PDE	Partial differential equation(s).	Ch. 1, Ch. 3
POD	Proper orthogonal decomposition; classical ROM method based on dominant modes of a correlation operator.	Ch. 1, Ch. 3
ROM	Reduced-order modeling; general family of low-dimensional surrogate models for high-dimensional evolutions.	Ch. 1
SR	Super-resolution; spatial upsampling from coarse to fine grids.	Ch. 3

Continued on next page

Abbrev.	Meaning	Where used
TDDFT	Time-dependent density functional theory; time-dependent extension of ground-state DFT.	Ch. 1
TDSE	Time-dependent Schrödinger equation; fundamental N-electron equation of motion.	Ch. 1, Ch. 4
VP	Vlasov–Poisson system; kinetic PDE for collisionless plasma dynamics.	Ch. 1, Ch. 3

Chapter 2

Learning nonlinear integral operators via recurrent neural networks and its application in solving integro-differential equations

Integro-differential equations (IDEs) arise in many scientific applications ranging from nonequilibrium quantum dynamics [168, 36, 142], the dynamics of non-Markovian colloidal particles, [211, 209, 214, 217], the modeling of dispersive waves, [191, 40] and electronic circuits [16]. One particularly important example is the application of the Kadanoff-Baym equations (KBE) [74, 168, 142] for the time evolution of quantum correlators. This equation describes the propagation in time of non-equilibrium Green's functions, and finding efficient methods of solving this equation is of extreme importance in the study of driven quantum systems. A general class of IDEs that the KBEs fall into can be written as:

$$\frac{d}{dt}G(t) = F(G(t),t) + \int_0^t K(G(t-s),s)G(s)ds \tag{2.1}$$

where both F and the integral kernel K are functions of and t and $G(t)$. This IDE is computationally challenging to solve due to the presence of the integral term in (2.1). A numerical time evolution scheme typically requires performing a numerical integration of the integral term at each time step. If n_T time steps are taken to evolve the numerical solution of (2.1) to time T, the overall computational complexity in time is proportional to at least n_T^2, which can be high for a large n_T. Several techniques have been recently

[0]The following chapter is based on published work in *Machine Learning with Applications* [7].

developed to reduce the temporal complexity of solving (2.1). One technique is based on an FFT-based solver that significantly speeds up the numerical integration of the integral term [80]. Another technique, known as dynamic mode decomposition (DMD), is a data-driven method that uses snapshots of the solution to (2.1) within a small time window to construct a reduced order model that can be used to extrapolate the long-time dynamics of $G(t)$ [199].

The main motivation of this research is to use machine learning approaches to reduce the computational complexity of solving IDE to the order of n_T. More specifically, by representing and learning the nonlinear integral operator $I(G(t),t) = \int_0^t K(G(t-s),s)G(s)ds$ in IDE (2.1) via neural network (NN), we turn an IDE into an ordinary differential equation (ODE) of the form

$$\frac{d}{dt}G(t) = F(G(t),t) + I(G(t),t), \tag{2.2}$$

where $I(G(t),t)$ is a functional of $G(t)$ and t that can be evaluated with a constant cost, i.e., without performing numerical integration. A standard ODE scheme can then be applied to solve (2.2) with a temporal complexity of $O(n_T)$.

In principle, the mapping from $G(t)$ to $I(G(t),t)$, which is implicitly defined by the integral $\int_0^t K(G(t-s),s)G(s)ds$ always exists, although its (memoryless) analytical form is generally unknown [181, 165]. In this work, we seek to represent and learn such a mapping by training a recurrent neural network (RNN) using snapshots of the numerical solution of (2.1) within a small time window. Hence our approach falls into the category of operator learning methods, which encompasses recently developed machine learning methodologies such as DeepONet [112] and the Fourier neural operator [101, 90]. In an operator learning method, we view the solution to a given ODE or partial differential equation (PDE) as the output of a neural network parameterized operator that maps between function spaces. For instance, solving an initial value problem for a given PDE can be reformulated as searching for a neural network parameterized operator, denoted as $S : u(x,0) \to u(x,t)$. Carefully designed neural networks, such as those employed in DeepONet and the Fourier neural operator, can approximate this operator S. By applying the learned operator S to the initial condition $u(x,0)$ we can readily obtain the solution to the PDE. One notable advantage of the operator learning approach lies in the generalizability of the trained neural network model. Once the approximation to the operator is obtained, it can be applied to other two initial conditions or inputs to the model.

In this work, we adopt an operator-learning perspective and choose to use a long-short term memory (LSTM)-based RNN [65] to learn the mapping between $G(t)$ and $I(G(t),t)$. Instead of a simple feed-forward neural network (FFNN), we utilize RNNs because they

can better preserve the causality of both $G(t)$ and $I(G(t),t)$. Furthermore, instead of learning the operator that yields the solution to the IDE (2.1) directly, we choose to learn the mapping between $G(t)$ and $I(G(t),t)$ for the following reasons. First of all, although the solution of (2.1) depends on both $F(G(t),t)$ (referred to as the *streaming term*) and $I(G(t),t)$ (referred to as the *memory integral* or *collision integral*), the cost of evaluating the memory term typically far exceeds that of the streaming term. Secondly, when the streaming term $F(G(t),t)$ is time-dependent and creates atypical dynamics outside of the training window (see e.g. Figure 2.6), it is difficult to directly learn a solution operator using short-time data. Consequently, the extrapolated prediction of the solution of (2.1) for large t can be poor. On the other hand, in many physically relevant scenarios, the integral kernel in (2.1) is well behaved. As a result, we expect the mapping between $G(t)$ and $I(G(t),t)$, which is independent of the solution $G(t)$ itself, can be learned more easily. Furthermore, in addition to increasing the time window and training the RNN with different initial conditions, we can augment the training data by considering solutions of (2.1) with different streaming terms within a small time window. We will refer to this type of training as multi-trajectory training in Section 2.2. This type of training is important for learning an operator that maps from one function space to another. Once the integral operator $I(t)$ is well approximated by a properly trained RNN, the solution of the IDE can be obtained using any standard ODE solver with $I(t)$ evaluated by the RNN. Furthermore, transferability of the learning algorithm is clearly achievable. Once the integral operator with a fixed integral kernel K is learned via an LSTM-based RNN, it can be used to solve the IDE (2.1) associated with different streaming forces $F(G(t),t)$ by using a standard ODE solver.

The approach presented in this work is similar in spirit to other developments in using machine learning (ML) techniques and neural networks (NN) to solve forward and inverse problems defined by ODEs and PDEs. The most prominent example is the physics-informed neural network (PINN) [32, 203] and its generalizations. In the PINN approach, an NN serves as a solver that takes the spatial-temporal coordinate x,t as the input and outputs the approximate solutions to the differential equation. The whole network is trained using the loss function that is defined in terms of the underlying differential equation. More recent members within the PINN family include sparse physics-informed neural network (SPINN) [140] and parareal physics-informed neural network (PPINN) [121]. Another representative approach is the neural ordinary differential equation (NeuralODE) [31] and it is particularly useful for solving the inverse problem of differential equations. In this approach, an NN is used to approximate the derivatives of the state variables, i.e. the "right-hand side" of a differential equation. After the recovery of the derivatives, solutions to the differential equations can be obtained using a standard numerical solver for ODEs and PDEs. Recent developments in

the NeuralODE family include NeuralSDE [108, 101], Neural Jump SDE [71], NeuralSPDE [152], and infinitely deep Bayesian neural networks [196]. What distinguishes the approach presented in this work from other existing neural-network-based differential equation solver is that we do not use the RNN to solve the IDEs directly. Instead, we use it to approximate the time dependent nonlinear integral operator that is costly to evaluate. We combine the RNN based operator learning with a standard ODE solver to obtain numerical solutions to the IDEs. To demonstrate the efficiency of this method, we will benchmark our result against a direct operator learned model that learns the entire solution to the right hand side of (2.2), as well as against using DMD to extrapolate the dynamics.

This chapter is organized as follows. In Section 2.1, we first introduce the basic architecture of RNN and the LSTM model, then we customize a specific RNN model designed to approximate the integral functional $I(t)$ and introduce two training methods for the RNN model. In Section 2.2, we use a nonlinear, complex-valued IDE for 2×2 matrix $G(t)$ as an example to demonstrate the effectiveness of using RNN to learn the integral operator and how this can be used to extrapolate the dynamics of the IDE. Importantly, we showcase the remarkable ability of using the same integral operator to solve IDEs driven by different streaming terms, thus affirming its generalizability. In Section 2.3, we will consider a specific physical application of the introduced methodology. Here, we underscore the practical utility of the RNN model in solving the Dyson's equation, offering further insights into its applicability. The primary findings and conclusions of this chapter are succinctly summarized in Section 2.4.

2.1 Methods

2.1.1 Recurrent neural networks

Recurrent neural networks (RNN) and their various adaptations have become prevalent tools to analyze and forecast the patterns in time series data [204, 79, 67] across a wide range of domains and scenarios [34, 25]. In this section, we will introduce the basic RNNs model, as well as the LSTM model.

2.1.2 Basic RNN model

The workflow of the RNN model is illustrated in Figure 2.1 (Top), which comprises several key components: including inputs, hidden states, and an RNN cell. The hidden states are calculated recursively within the RNN cell, utilizing sequential inputs. These hidden states play an important role in capturing temporal dependencies and facilitating

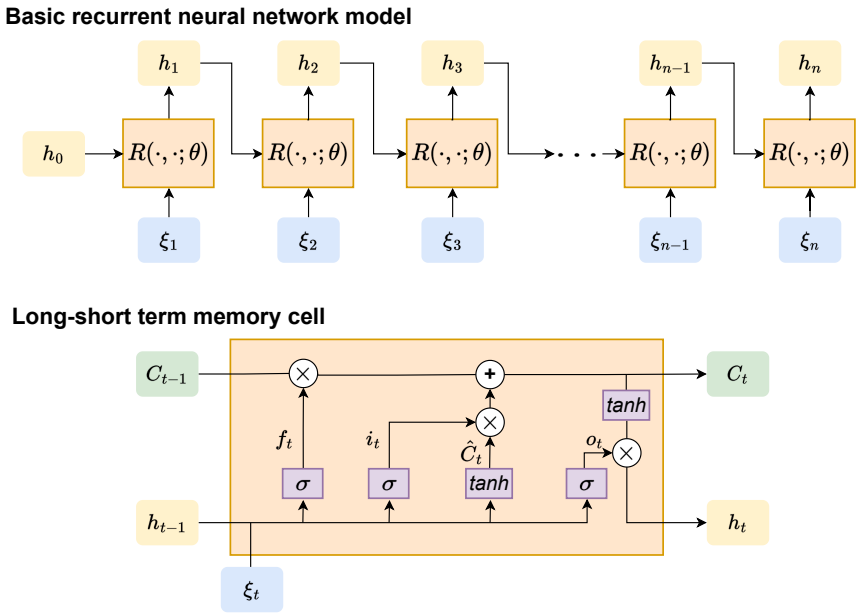


Figure 2.1: **(Top) The workflow of a basic RNN model.** The model processes the inputs $\{\xi_1, \xi_2, \cdots, \xi_n\}$ one by one in sequence and produces a corresponding sequence of hidden states $\{h_1, h_2, \cdots, h_n\}$. The same set of model parameters, θ, is employed in the RNN cell (i.e., $R(\cdot,\cdot;\theta)$) to calculate each h_i, $i = 1, \cdots, n$. **(Bottom) An LSTM cell**. The update of the cell state C_t and the hidden state h_t relies on the current time input ξ_t and the preceding states, h_{t-1} and C_{t-1}, through several information gates.

the propagation of information over time within sequential data. Furthermore, these hidden states can be employed to model time-dependent quantities of interest, such as the memory integral $I(G(t), t)$, which is the main focus of our chapter.

Specifically, consider a time series dataset $\Xi = \{\xi_0, \xi_1, \xi_2, \ldots, \xi_N\}$, where ξ_i represents the i^{th} time index within our time series. The RNN cell, represented as a function $R(\cdot,\cdot;\theta)$, takes the preceding hidden state and the current time data as input and produces a new hidden state as output. Namely, $h_t = R(\xi_t, h_{t-1};\theta), t = 1, \cdots, N$. Here, h_0 is initialized as a zero vector and θ represents the set of shared and trainable parameters. Mathematically, the hidden states can be written as a composition of the RNN cells given by:

$$h_n = R(\xi_n, h_{n-1};\theta) = R(\xi_n, R(\xi_{n-1}, h_{n-2};\theta);\theta) = R(R(...R(\xi_2, R(\xi_1, h_0;\theta);\theta), ...;\theta);\theta).$$

The sharing of θ enables the RNN to learn general patterns across time and incorporate a memory effect into the model. This capability also allows RNN models to handle input sequences of varying lengths.

2.1.3 Long-short term memory (LSTM)

The LSTM model is a type of RNN that distinguishes itself by employing different gates within its LSTM cell to regulate the flow of information, as shown in Figure 2.1 (Bottom). Compared with the standard RNN models, LSTM models exhibit better performance in capturing long-range dependencies and mitigating the vanishing gradient issue [65]. Their gated structure allows them to effectively preserve information across longer sequences. As a result, LSTM models have gained widespread applications in dynamics modeling tasks [213, 61]. In our specific context, we will employ an LSTM model to model the memory integral of the IDE.

The LSTM cell features two distinct states: the cell state, denoted as C_t, and the hidden state, represented as h_t. The cell state serves as a repository for long-term memory, capable of retaining and propagating information throughout different time steps. On the other hand, the hidden state captures current information, serving as the foundation for predictions made at each time step. To regulate the flow of state information, three pivotal gates are employed: the input gate, the forget gate, and the output gate. The forget gate is responsible for determining which information from the previous cell state should be retained in the current cell state, while the input gate governs the incorporation of new information into the current cell state. Concurrently, the output gate controls the outward transmission of information from the cell state.

Let i_t, f_t, and o_t represent the input, forget, and output gates at time t. Each of these gates employs a nonlinear activation function, such as $\sigma(x) = \frac{1}{1+e^{-x}}$, in conjunction with learnable parameters to govern the selection of information to be incorporated into the state updates. This can be expressed as follows:

$$i_t = \sigma(\xi_t, h_{t-1}; \theta_{input}), \quad f_t = \sigma(\xi_t, h_{t-1}; \theta_{forget}), \quad o_t = \sigma(\xi_t, h_{t-1}; \theta_{output}).$$

Here, θ_{input}, θ_{forget}, and θ_{output} are parameter sets within total collection of shareable parameters θ.

Using these gates, the current cell state is updated according to:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \hat{C}_t,$$

which combines both long-term memory (e.g., C_{t-1}) and new information (e.g., ξ_t used in the computation of $\hat{C}_t = tanh(\xi_t, h_{t-1}; \theta_c)$). Here, $\theta_c \in \theta$ is the parameter set. Finally, we compute the current hidden state h_t as:

$$h_t = o_t \cdot \tanh(C_t).$$

The hidden state h_t can subsequently undergo further trainable transformations, such as a linear transformation, to predict the specific quantity of interest.

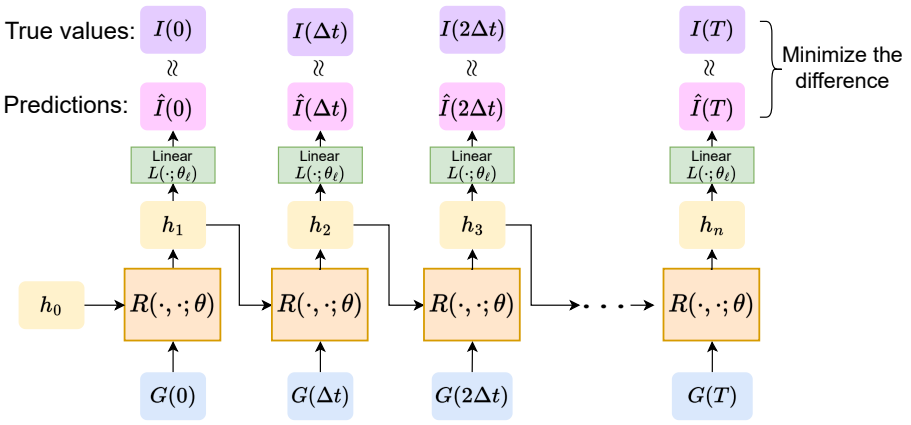


Figure 2.2: Learning and prediction diagram of the RNN model. In the training phase, the input sequence $\{G(0), G(\Delta t), G(2\Delta t), \ldots, G(T)\}$ is fed into the model to generate predictions $\{\hat{I}(0), \hat{I}(\Delta t), \hat{I}(2\Delta t), \ldots, \hat{I}(T)\}$ using the parameters θ. From here, we can use the final output $\hat{I}(T)$ to produce numerically extrapolated dynamics using the procedure outlined in *Model Architecture* and illustrated in Figure 2.3.

2.1.4 RNN customized for learning time-dependent nonlinear integral operator

Having introduced the basic architecture of RNN and the integro-differential equation we aim to solve, in this section, we customize a specific RNN model that enables us to efficiently learn the integral operator of an IDE within a short time window and use that to predict its long-time dynamics. As mentioned in the introduction, the RNN we seek should learn a map $I : G(t) \to I(t)$ for any given function $G(t)$. Since $I(t)$ depends on all the history values of $G(t)$, to capture this memory effect, we propose to use the LSTM cells as the basic modeling modules to build the RNN model. This leads to a *learning-predicting* diagram as illustrated in Figure 2.2.

2.1.5 Model architecture and dynamics extrapolation

The RNN in Figure 2.2 consists of an LSTM model and a linear transformation layer attached to it that maps the output of LSTM cells into the shape of the target time series. For our model, the NN takes the discretized $G(t)$ in a time grid $t = i\Delta t$ as the input and produces an approximated solution of the collision integral $\hat{I}(t)$ for $t = i\Delta t$ as the output. The RNN model is trained using a sufficiently accurate numerical solution to the IDE within a time window $t \in [0, T]$. After the training is done, in the extrapolation

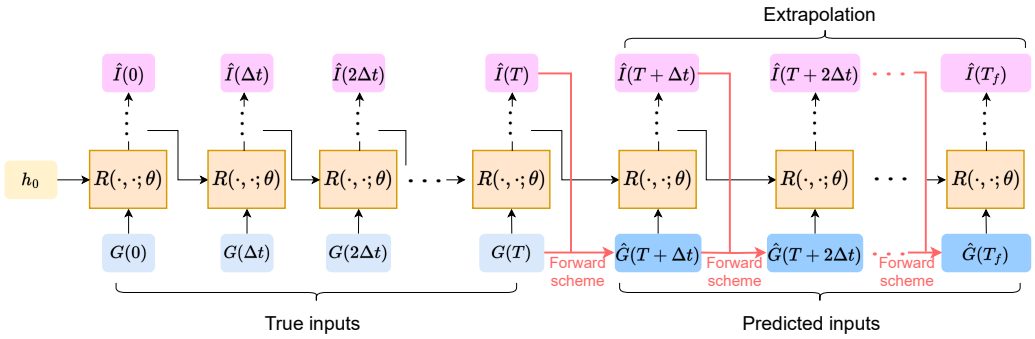


Figure 2.3: Extrapolation of dynamics using the RNN model. The training phase will output predictions $\{\hat{I}(i\Delta t)\}_{i=0}^{T}$. Using $\hat{I}(T)$, we can use a forward numerical method to obtain the numerically extrapolated $G(T+\Delta t)$. From here, we can recursively feed the numerically extrapolated $G(T+\Delta t)$ into the model to obtain the prediction for $\hat{I}(T+\Delta t)$. We repeat this until the final time, T_f.

phase, we can use an efficient and accurate forward propagation scheme to generate long-time trajectories. As an example, if we employ the forward Euler scheme to solve the differential equation, then for IDE (2.2), we have

$$G(T+(i+1)\Delta t) = G(T+i\Delta t) + \Delta t[F(T+i\Delta t, G(T+i\Delta t)) + \hat{I}(T+i\Delta t)]$$

where $\hat{I}(T+i\Delta t)$ is the output of the RNN generated recursively by feeding in the input $G(T+i\Delta t)$ as illustrated in Figure 2.3. The forward scheme for any multi-step method can be similarly derived. The specific model parameters, such as the hidden size of each layer, will be discussed in Section 2.2 per IDE considered. In this work, we utilize a PyTorch backend to implement all of our NN models.

2.1.6 Data preparation and loss functions

To generate the training data, we solve the IDE numerically using an Adams-Bashforth third-order method (AB3) and Simpson's rule to approximate the collision integral $I(t)$. This generates a time series data that is recorded in an interval of $[0,T]$, discretized by Δt, eventually forming a dataset consisting of $\{(G(i\Delta t), I(i\Delta t)\}_{i=0}^{T}$ pairs. For the IDEs with complex values that are considered in our applications, we will further split the real and imaginary parts of $G(t)$ and $I(t)$ when we generate the input sequence and compare $\hat{I}(t)$ with $I(t)$. This has to be done since most popular ML frameworks such as PyTorch [134] natively support and are optimized for real-valued calculations. The parameters of the network, denoted by θ, are randomly initialized. The Adam optimizer [85], which is a

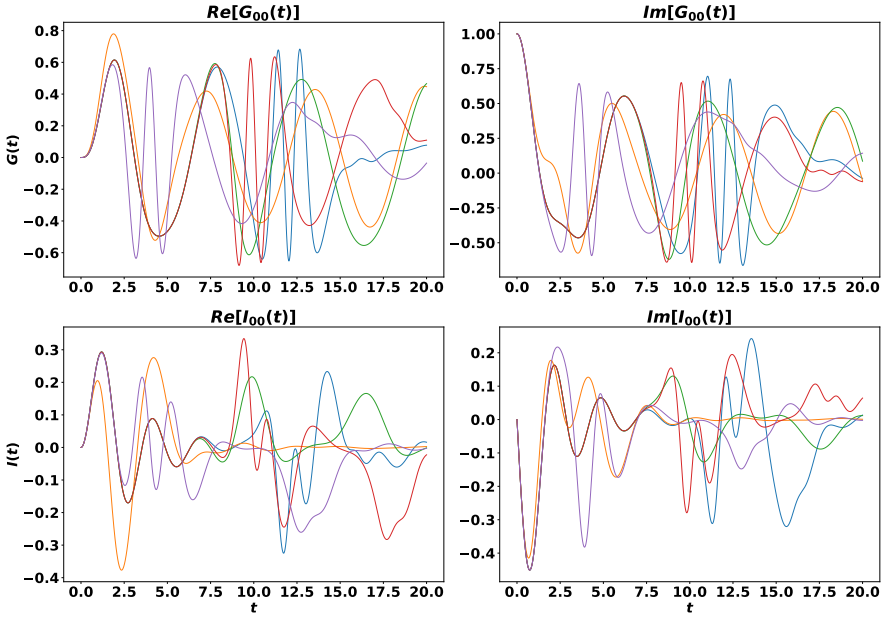


Figure 2.4: Sample trajectories of $G_{00}(t), I_{00}(t)$, selected from the database created by solving (2.4) with $\alpha_1, \alpha_2 \in [1, 20] \times [1, 20]$, $\sigma \in \{1, 2, 3, 4, 5\}$, and $\beta = 1$. Subsequently, we use the same database to do multi-trajectory training of the RNN model.

first-order method that uses gradient-based optimization, is chosen to adjust the parameter values to minimize the mean-squared error (MSE) function:

$$f(I, \hat{I}; \theta) := \frac{1}{N} \sum_{i=0}^{N} (I(i\Delta t) - \hat{I}(i\Delta t))^2, \tag{2.3}$$

where $\hat{I}(i\Delta t)$ is the predicted collision integral generated by the RNN and $I(i\Delta t)$ is the ground truth. The parameters are learned by propagating the gradients of each hidden state's inputs.

2.1.7 Training method

For all the numerical simulations considered in this chapter, the RNN training is performed in a small time window (T sufficiently small). The trained model is then used for long-time (T_f) extrapolation in which $T_f \gg T$, (see Figure 2.3). We employ two training strategies:

1. (*Single trajectory training*) For this case, the RNN is trained using a *single* trajectory dataset $\{(G(i\Delta t), I(i\Delta t)\}_{i=0}^{N}$. The numerical advantages of employing single trajectory training stem from its relatively low computational cost throughout the optimization process, as the dataset consists of a single trajectory. Accordingly, since the single trajectory data corresponds to a *specific* choice of the streaming term $F(G(t), t)$, the optimized RNN normally yields bad *generalization* results to solve the same IDE with different $F(G(t), t)$ terms.

2. (*Multi-trajectory training*) In contrast with the first case, the RNN can also be trained using batch training techniques, where the input becomes *multiple* trajectories generated by choosing different streaming terms $F(G(t), t)$. The training cost is obviously higher but the obtained RNN model has greater generalizability and can be used to predict dynamics for the IDE with new streaming terms. From the operator-learning point of view, the multi-trajectory training method is preferred since the enlarged dataset contains different input-output $G(t)$ and $I(t)$, which provides more samples for learning the mapping $I : G(t) \to I(t)$ for a given IDE.

In accordance with these two training strategies, we also use two validation methods to detect and avoid overfitting. For the single trajectory training case, we can split the dataset $\{(G(i\Delta t), I(i\Delta t)\}_{i=0}^{T}$ into two parts: $\{(G(i\Delta t), I(i\Delta t)\}_{i=0}^{K}$ and $\{(G(i\Delta t), I(i\Delta t)\}_{i=K+1}^{T}$, and then using the first time series to measure training loss used for optimization of the RNN and the second part to measure the validation error. For the multiple trajectory training, the validation error is calculated using a new dataset $\{(G(i\Delta t), I(i\Delta t)\}_{i=0}^{T}$ obtained by solving the IDE with streaming terms $F(G(t), t)$ that are different from those ones in the training dataset. The minimized validation error normally indicates the generalizability of the obtained RNN model. For the first case, the generalizability is reflected in the predictability of long-time dynamics. For the second case, it is also reflected in the predictability of the dynamics for IDE with new streaming terms.

The advantage to the single trajectory training scheme is that a low amount of computational resources is needed to generate the training data for the LSTM-RNN model since we are only learning a functional for a specific trajectory. This, however, comes with the tradeoff that the streaming term $F(G(t), t)$ in (2.2) can not be changed arbitrarily, and must match closely to the training trajectory chosen. In comparison, the multi-trajectory training scheme has a larger computational resource requirement to train the NN. However, since we use multiple streaming terms $F(G(t), t)$ for the training data, the trained RNN is able generalize to new kinds of streaming terms post-training.

2.1.8 Computational cost

All computations are performed using the Perlmutter cluster. The login node has one AMD EPYC 7713 as the CPU and one 40GB NVIDIA A100 as the GPU. For (2.4), a typical training with input sequence length 2000 across 750 epochs for an RNN with 2 LSTM layers and hidden size 64 would take approximately 2 hours for the multi-trajectory training and 30 minutes for the single trajectory training. Under the same setting, for (2.5), it would take approximately 30 minutes for a multi-trajectory training and approximately 15 minutes for a single trajectory training. After the training, in the extrapolation phase, the computational time would be spent on the evaluation of the multi-step forward scheme and for RNN to generate new output $I(T+i\Delta t)$. Detailed runtime statistics for the numerical examples considered in this chapter will be tabulated in the Results section. Generally speaking, since the RNN is made of multi-fold function compositions, the generation of the output sequence is immediate. This ensures the total computational cost in the extrapolation phase scales as of $O(n_T)$, in contrast with the $O(n_T^2)$ scaling for a normal IDE numerical solver. As a result, the entire computational cost is greatly reduced for long-time simulations.

2.2 Numerical results

To demonstrate our method, we first consider a nonlinear complex-valued IDE given by:

$$\frac{d}{dt}G(t) = A(t)G(t) + \int_0^t K(t-s)G(s)ds. \tag{2.4}$$

Here we take:

$$A(t) = -i\begin{bmatrix} -\beta e^{\frac{-(t-\alpha_1)^2}{\sigma}} & 1 \\ 1 & \beta e^{\frac{-(t-\alpha_2)^2}{\sigma}} \end{bmatrix}, \qquad G(0) = \begin{bmatrix} i & 0 \\ 0 & i \end{bmatrix}, \qquad K(t) = \cos\left[\frac{G^2(t)}{4}\right].$$

For (2.4), we will employ both the single trajectory training and multi-trajectory training approaches to approximate the integral operator $I(t) := \int_0^t K(t-s)G(s)ds$ and then combine the learned $I(t)$ and AB3 as the ODE solver to obtain long-time trajectories. Before we present the training details, we note in advance that the RNN results are benchmarked in three ways:

1. We compare the learned and extrapolated $G(t)$ to a numerical solution of (2.4) using AB3 + Simpson's rule as the numerical solver. We then show the accuracy in the training window and ability to extrapolate future dynamics outside of the training

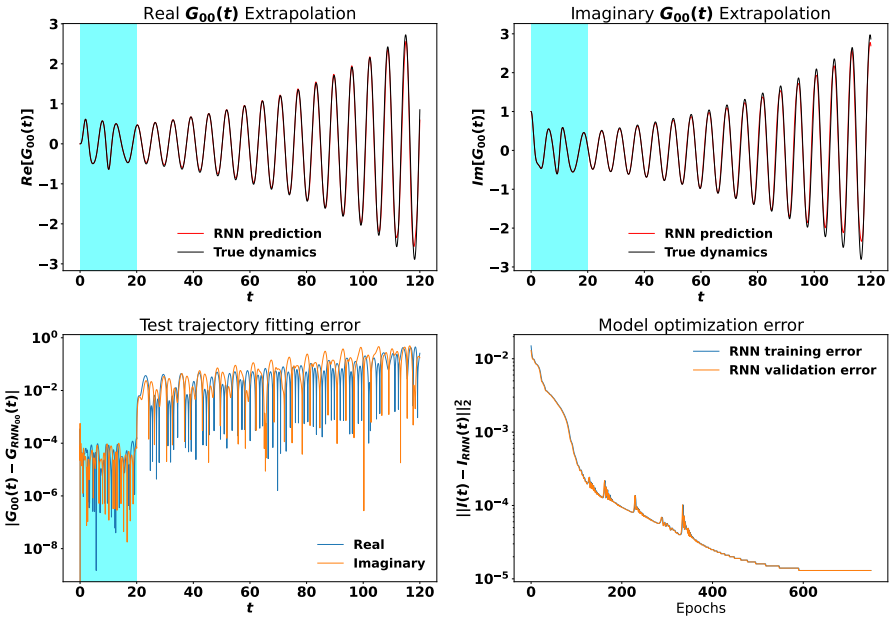


Figure 2.5: Single trajectory learning of (2.4) where the RNN is trained on $\alpha_1 = 10, \alpha_2 = 15$, $\sigma = 2$ and $\beta = 1$ and tested by extrapolating the dynamics up to $T = 120$ into the future. The shaded window represents the training regime. The black curve represents the true dynamics. The red curve represents the simulated dynamics using the learned $\hat{I}(t)$ produced by the RNN.

window. Due to the fact that the RNN is designed to learn the map $I : G(t) \to I(t)$, we also expect that extrapolating dynamics outside the training window to be generally valid in the multi-trajectory case for (2.4) with different parameterizations of the $A(t)$ term.

2. We further compare our learning strategy, i.e. learning the map $I : G(t) \to I(t)$, with two dynamics extrapolation methods. First, we consider a direct learning strategy that uses an RNN to directly predict the next timestep $G((i+1)\Delta t)$, given $G(i\Delta t)$. Accordingly, we modify the loss function (2.3) as:

$$f(G, G_{RNN}; \theta) := \frac{1}{N} \sum_{i=0}^{N} (G(i\Delta t) - G_{RNN}(i\Delta t))^2.$$

and keep the same RNN architecture as before. Second, we also compare our results with what was obtained using dynamic mode decomposition (DMD) [154, 155, 92, 199, 200].

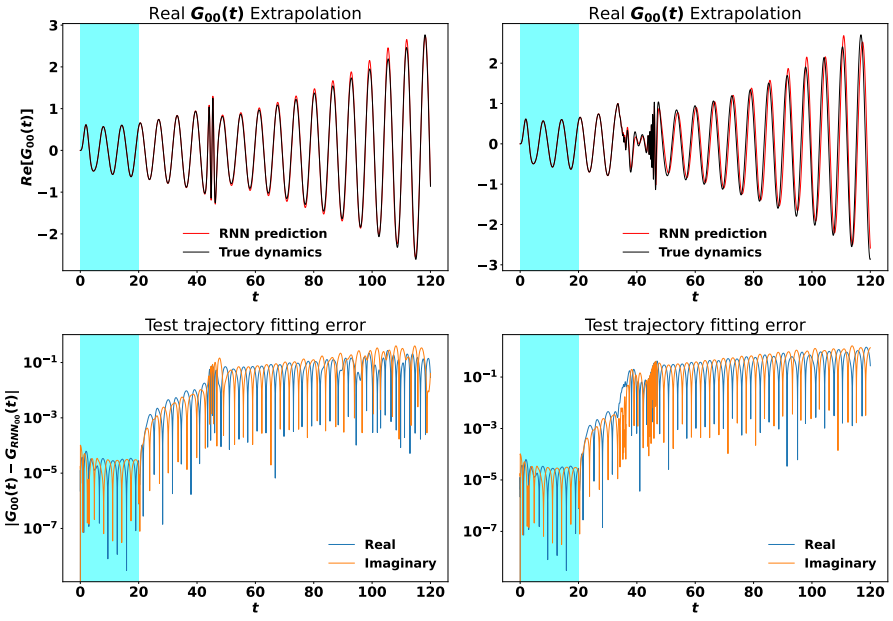


Figure 2.6: Multi-trajectory learning of (2.4) where the RNN is trained on $\alpha_1, \alpha_2 \in [1, 20] \times [1, 20]$, $\sigma \in \{1, 2, 3, 4, 5\}$, $\beta = 1$ and tested on $\alpha_1 = 45, \alpha_2 = 45$, $\sigma = 5$ and $\beta = 1$ (**First column**) and $\alpha_1 = 45, \alpha_2 = 35$, $\sigma = 2$ and $\beta = 14$ (**Second column**). The shaded window represents the training regime. The black curve represents the true dynamics. The red curve represents the simulated dynamics using the learned $\hat{I}(t)$ produced by the RNN model.

3. We calculate the wall-clock runtime of our simulation and compare it with what was obtained using a standard IDE solver. This would demonstrate the numerical speedup we gain using the RNN to approximate the memory integral in the IDE.

The weaknesses of the benchmark methods come from the fact that these two approaches rely heavily on the training data used to predict future timesteps. For instance, since DMD constructs the model based on a small window of the dynamics within a specific time frame to extrapolate the future dynamics, if we are to observe a streaming term in (2.2) that has forcing applied *outside* of the training window, the method will fail simply because the training data lacking in this aspect. In the same line of reasoning, we also expect the direct model to fail in this case.

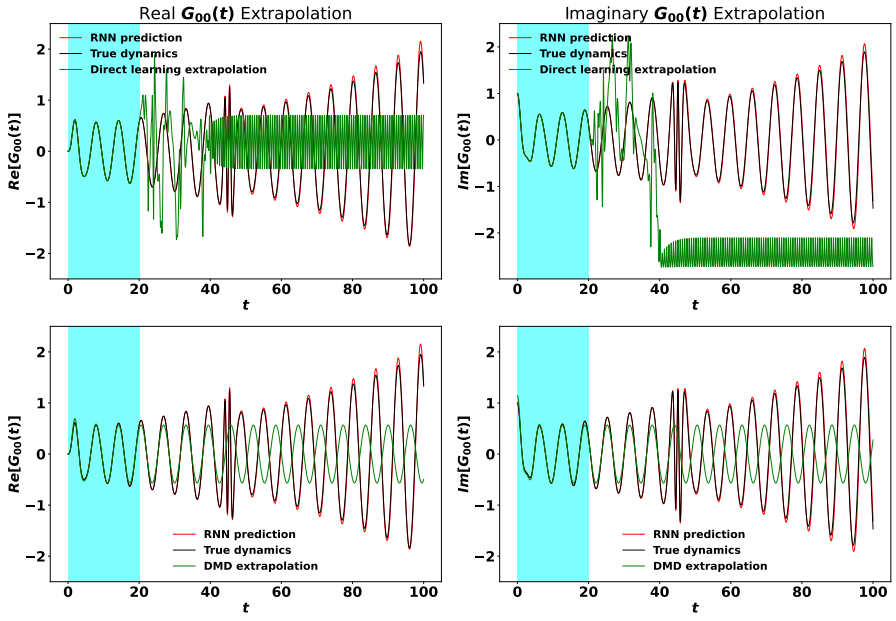


Figure 2.7: Multi-trajectory learning of (2.4) where the RNN is trained on $\alpha_1, \alpha_2 \in [1, 20] \times [1, 20]$, $\sigma \in \{1, 2, 3, 4, 5\}$, $\beta = 1$ and tested on $\alpha_1 = 45, \alpha_2 = 45$, $\sigma = 5$ and $\beta = 1$. The shaded window represents the training regime. The black curve represents the true dynamics. The red curve represents the simulated dynamics using the learned $\hat{I}(t)$ produced by the RNN model. The green curve in the top row represents the simulated dynamics using the RNN model that directly predicts $G(t)$. The green curve in the bottom row represents the DMD extrapolation result.

2.2.1 Training details

For single trajectory training, we fix the modeling parameters in (2.4) to be $\alpha_1 = 10, \alpha_2 = 15, \sigma = 2$, and $\beta = 1$. The RNN is trained by feeding in the singular trajectory $\{G(i\Delta t)\}_{i=1}^{N}$ for $\Delta t = 0.01$ and $N = 2000$. This means the training window is until time $T = 20$. Then, the extrapolated trajectory of $G(t)$ is generated up to $T = 120$ ($N = 10000$ timesteps into the future).

For multi-trajectory training, the dataset used for training is generated by solving (2.4) with different parameters $\alpha_1, \alpha_2, \sigma, \beta$. Specifically, we prepare a dataset by choosing α_1, α_2 from the lattice grid $[1, 20] \times [1, 20]$, $\sigma \in \{1, 2, 3, 4, 5\}$ and $\beta = 1$. This results in 2000 different trajectories. We then batch these 2000 trajectories together using a batch size of 128. In turn, for each epoch of training, every set of batched data is the fed to the RNN for training. Similar to the single trajectory case, each trajectory in the multi-trajectory

dataset is recorded until time $T = 20$ for training.

Plotted in Figure 2.4 is a reference of some of our input data $G(t)$ and output data $I(t)$. The displayed result is the first component of the 2 x 2 matrices $G(t)$ and $I(t)$. The dynamic difference between different matrix components is minimal.

The RNN model contains 2 LSTM layers where the hidden size for each layer is 64. The input is an 8-dimensional vector that consists of the flattened 2×2 matrix $G(t)$ decomposed into the real and imaginary parts. Similarly, the output is 8-dimensional, consisting of the real and imaginary parts of the 2×2 matrix collision integral $I(t)$. The Adam optimizer has an initial learning rate set to be 0.01, with an adaptive cosine learning rate that decays accordingly with respect to total epochs used for training. This is designed to help us converge closer to the optimal solution as we progress in our optimization by taking smaller steps. The RNN is trained over 750 total epochs.

2.2.2 Results discussion

The results for extrapolating long-term dynamics using our scheme are summarized in Figures 2.5-2.7. The single trajectory result is displayed in Figure 2.5. We see that our RNN scheme is able to accurately predict the long-term dynamics of the system using approximately 16% of the total trajectory for training. In particular, the modes and amplitudes of the oscillations match well with the ground truth dynamics.

The multi-trajectory training results are shown in Figure 2.6. In the first column of Figure 2.6, the learned integral operator $I(t)$ is used to solve (2.4) for a new set of parameters: $\alpha_1 = 45, \alpha_2 = 45, \sigma = 5$, and $\beta = 1$, which is *outside* of the parameter range of the training dataset. It is to be noted that by choosing $\alpha_1 = 45$ and $\alpha_2 = 45$, the Gaussian pulse from the $A(t)$ term in (2.4) is applied at time $T = 45$. This is far outside of the training window $T \in [0, 20]$. We see that our RNN model is still able to accurately predict the dynamics of this test trajectory, despite the highly oscillatory regime of the test trajectory appearing much later in time. This result is further highlighted in the second column of Figure 2.6 where the test trajectory corresponds to $\alpha_1 = 45, \alpha_2 = 35, \sigma = 5$, and $\beta = 14$, which creates a large chirp oscillation far after the training window. The predicted dynamics using our method still matches extremely well with the ground truth dynamics. These two test examples clearly demonstrate the generalizability of using the RNN model to model the integral operator of (2.4) for predicting long-term dynamics.

Our results are compared against the two benchmark methods mentioned previously: the direct learning approach and DMD. The results are summarized in Figure 2.7. Here, we test both methods on a test trajectory corresponding to parameters $\alpha_1 = 45, \alpha_2 = 45$, $\sigma = 5$ and $\beta = 1$ in (2.4) until time $T = 20$. As we introduced before, the direct learning

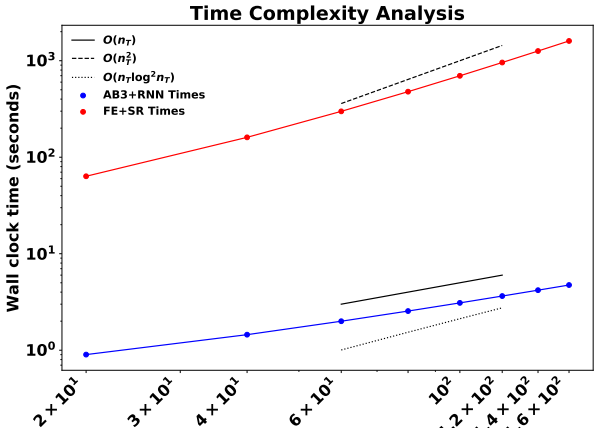


Figure 2.8: Scaling limit of the wall-clock run time of different approaches for solving (2.4). The horizontal axis represents the queried extrapolation length for a trajectory.

approach used the same RNN architecture and multi-trajectory training dataset for training to predict $G(i\Delta t) \to G((i+1)\Delta t)$. Since DMD can only do time extrapolation for a single trajectory at a time, the single test trajectory above is used for dynamics extrapolation. Both methods are then used to extrapolate the dynamics until time $T = 120$. From Figure 2.7, it is evident that the integral operator learning strategy significantly outperforms the other approaches. This arises from the fact that DMD and the direct learning method are both tested on a trajectory that has a streaming term that perturbs (2.4) *outside* of the training window.

Lastly, we comment on the computational cost reduction brought by the RNN integral operator learning. In Table 2.1, we record the wall clock runtimes used to generate different lengths of $G(t)$ in the extrapolating regime for our approach and a regular IDE solver. As we mentioned previously, the AB3+RNN is used to generate the extrapolated trajectory. For comparison, we employ an IDE numerical solver which consists of an Euler scheme for time integration, and Simpson's Rule for approximating the collision integral $I(t)$ (FE+SR). We see from Table 2.1 that the computational cost of querying the RNN and running AB3 is significantly lower than running FE+SR. Moreover, the overall scaling limit of our approach is of the order $O(n_T)$, in contrast with the $O(n_T^2)$ using FE+SR,and $O(n_T \log^2 n_T)$ using the method from a recent work [80]. The results are visualized in Figure 2.8. Our approach has $O(n_T)$ scaling essentially because querying an RNN on a length T sequence has cost $O(n_T)$. In conjunction with any forward solver, such as AB3, we see that the total scaling will be $O(n_T)_A + O(n_T)_B = O(n_T)$, where $O(n_T)_A$ results from the numerical method, and $O(n_T)_B$ results from querying the RNN.

In contrast, using a numerical integration for the integral term and a forward solver to solve IDE (2.1) has $O(n_T^2)$ scaling due to the integral term needing to be computed at each time step for all the history. We also note that for both the RNN and the FE + SR methods in Table 2.1 use $\Delta t = 0.01$.

T_f **(final time)**	**AB3 + RNN (wall-clock s)**	**FE + SR (wall-clock s)**
20	0.8684	60.3825
40	1.5028	166.0921
80	2.5220	475.0468
160	4.7412	1602.8664

Table 2.1: Comparison of wall-clock time using the RNN method and a regular IDE solver to extrapolate dynamics with $\Delta t = 0.01$ for both methods. Here, total simulation time refers to the final physical time T_f used in solving the IDE.

2.3 Application in quantum dynamics simulation

In this section, we use our RNN method to numerically solve and extrapolate the dynamics of Dyson's equation, which is a special kind of complex IDE that is fundamentally important for the study of quantum many-body systems [168]. Dyson's equation is used extensively in quantum many-body systems to describe the interaction between particles and their environment. This aids in understanding phenomena such as particle scattering and vacuum fluctuations.

To benchmark our result, we consider an equilibrium Dyson's equation for hopping electrons in the Bethe lattice [80, 113]. When $t > 0$, the equation of motion reads:

$$i\partial_t G^R(t) = hG^R(t) + \int_0^t c^2 G^R(t-s)G^R(s)ds. \tag{2.5}$$

We see that (2.5) falls nicely into the class of IDEs (2.1) that we sought out to study. In the context of quantum many-body theory, the time-dependent quantity $G^R(t)$ is called the retarded Green's function. $G^R(t)$ contains important physical information of the system such as the single-particle energy spectrum. The memory kernel $K(t-s) = c^2 G^R(t-s)$ is the self-energy of the system. Here, h, c are the modeling parameters that will be varied when we do multi-trajectory training. Throughout this section, the initial condition

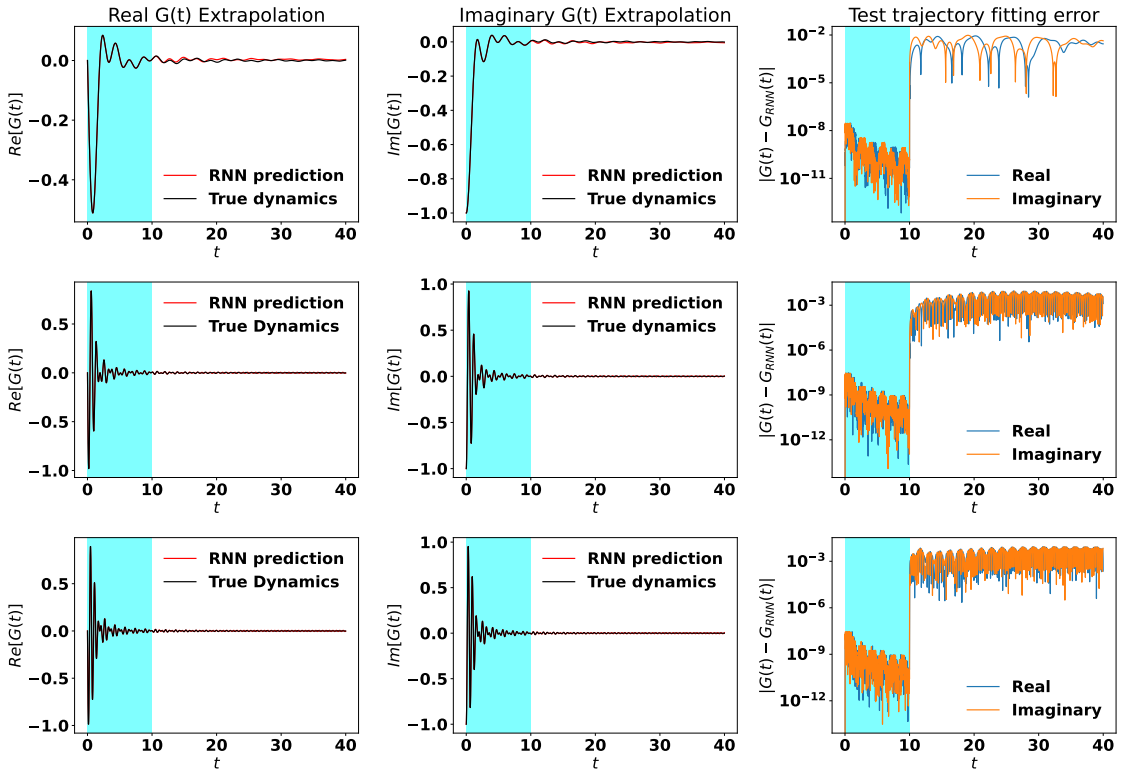


Figure 2.9: RNN training and extrapolating results for (2.5). The first row displays the single trajectory training result for $h = -1, c = 1$. The second row is the multi-trajectory training results where the RNN is trained for $h \in [1, 10], c = 1$, and tested on $h = 8, c = 1$. The third row uses the same multi-trajectory model, but tests on $h = 11.5, c = 1$.

$G^{\mathrm{R}}(0) = -i$ is chosen. According to the analysis by Kaye et al. [80], (2.5) admits the analytical solution:

$$G^R(t) = -ie^{-iht}\frac{J_1(2ct)}{ct}, \qquad t > 0,$$

where $J_1(t)$ is the Bessel function of the first kind. We will use this analytical solution to benchmark the extrapolation results generated by RNN.

2.3.1 Training details

For (2.5), we use an RNN model with 2 layers of LSTM cells, with the hidden state size 128 for each layer. The input and output are the same as it was for (2.4), where

the solution $G^R(t)$ is fed as input to the RNN to learn the corresponding output $I(t) = \int_0^t c^2 G^R(t-s)G^R(s)ds$. For the single trajectory training case, we set $h = -1$ and $c = 1$. For the multi-trajectory training, the interval $[1, 10]$ is evenly spaced into 2000 points. We take h from this discretized set while keeping $c = 1$. This yields a total of 2000 trajectories as the dataset for the subsequent RNN training. We then train the RNN in the same fashion as in Section 2.2.

2.3.2 Results discussion

All of the results are summarized in Figure 2.9. In the first row, we show the single trajectory training result where the input data is collected until time $T = 10$ and then extrapolated until time $T = 40$ (3000 timesteps into the future for $\Delta t = 0.01$). The second row shows the multi-trajectory training results where the parameters for the test trajectory are set to be $h = 8, c = 1$. Note that h is chosen from the sampling domain $[1, 10]$ but *is not* chosen as a parameter to be used in the training dataset. The third row shows the multi-trajectory training results where the parameters for the test trajectory are set to be $h = 11.5, c = 1$, which lies outside of the training dataset completely.

Both the single trajectory and the multi-trajectory results yield precise predictions of the future-time dynamics of Dyson's equation. All these findings are consistent with the previous results for (2.4). In comparison, we also see more accurate and robust results in the multi-trajectory case, both visually and in terms of the error plots. We speculate that this is the case because the dynamics exhibit a more tame limiting behavior as $t \to \infty$ as compared to (2.4), in conjunction with (2.5) being a scalar-valued IDE compared to a matrix-valued IDE. We find in all cases that our RNN method makes accurate predictions for the dynamics well past the training window.

2.4 Conclusion

In this chapter, we introduced an RNN-based machine learning technique that uses LSTMs as the basic modeling module to learn the nonlinear memory integral term in a class of IDEs. Such a learning scheme allowed us to turn a computationally challenging IDE into an ODE that can be solved efficiently using standard ODE solvers. We showed that a more effective way to learn the nonlinear integral operator is to include multiple training trajectories defined by different streaming terms in the IDEs considered. The generalizability of the learned operator was demonstrated by using the learned map to predict the dynamics of trajectories that are driven by completely different streaming terms outside of the parameter range of the training data. Moreover, since the RNN

consists of layers of function composition, we showed that it is almost immediate to generate a next timestep collision integral $I(t)$ given the input. This led to an overall $O(n_T)$ scaling of computational cost (the same as an ODE solver) when we used the RNN as a proxy for the integral term in the IDE, in contrast with the $O(n_T^2)$ scaling of a regular IDE solver. Due to the scalability of the RNN architecture, we expect that the methodology can be generalized and used to solve high-dimensional IDEs, such as the Kadanoff-Baym equations in nonequilibrium quantum many-body theory. This was accomplished by collaborators in [215].

Chapter 3

SRaFTE: Super-Resolution and Future Time Extrapolation for spatiotemporal dynamics with applications to electron dynamics

3.1 Introduction

Partial differential equations (PDEs) form the mathematical backbone for a variety of problems in the physical sciences and engineering. Reliable numerical solutions, therefore, underpin progress in areas as diverse as climate forecasting [185], photonics [128], and materials design [169]. A persistent obstacle in being able to generate solutions to these PDEs, however, is the scale at which their dynamics evolve [97, 194]. This is closely tied to *resolution*; important physical effects for a system of interest often emerge only after one refines the resolution of the computational grid to be fine enough. Performing such a refinement, however, proves to be costly even with modern day supercomputing power.

A natural response to this computational bottleneck has been the emergence of machine learning (ML) based accelerations for PDE solvers [19, 20], which can be broadly grouped into two complementary strands. The first strand projects the full high-dimensional dynamics onto a reduced set of variables, yielding a lower-order evolution equation that often includes an unresolved closure term representing the missing subgrid-scale physics. A neural network is then trained to approximate this term,

[0]The following chapter is based on submitted work to *APL Computational Physics* [8].

thereby retaining the structure of the governing equations while supplying the missing physics that simplified models lack [56, 47, 18, 12, 7, 215]. The second strand adopts a fully equation-free, data-driven approach, opting to learn the complete propagator of the PDE and any refinements directly from training snapshots [109, 26, 111, 102, 122]. While this approach allows equation-agnostic training and inference, the lack of explicit physical constraints or governing equations can limit the fidelity and robustness of the learned propagators, especially when extrapolating the dynamics beyond the training regime.

In this chapter, we introduce a two-phase operator-learning framework—*Super-Resolution and Future-Time Extrapolation* (SRaFTE)—for learning and predicting PDE dynamics at fine scales. In Phase 1, we train a neural operator that maps coarse-grid PDE solutions to their fine-grid counterparts over short time windows. In Phase 2, the learned super-resolution operator is composed with a coarse-grid numerical integrator to construct an effective fine-grid propagator for the governing PDE. As such, the dynamics prediction procedure within SRaFTE can be understood as a predictor–corrector loop: at each step, the coarse-grid numerical solver advances the dynamics, and this coarse-scale prediction is then corrected by applying the super-resolution operator to lift it back to the fine grid. Because the coarse-grid integrator already respects the governing PDE and the neural operator serves only as a corrective lift, we expect the global approximation error to grow more slowly than in fully data-driven propagators or approaches that merely upsample predictions from the coarse-grid numerical solvers [68, 133, 30]. In this manner, SRaFTE has features of both strands of ML-based PDE acceleration framework and occupies a middle ground between fully equation-free and strictly PDE-constrained approaches. In particular, Phase 2 embeds the physical laws through the coarse-grid time integrator, whereas the learned super-resolution operator remains flexible and purely data-driven that does not explicitly enforce physical constraints at either resolution.

Furthermore, SRaFTE is not tied to a single data generation paradigm and can accommodate both (i) the commonly used restriction/synthetic-downsampling (sensor-style) setting [20, 91, 147, 146] and (ii) the simulation–to–simulation setting. In the former, coarse inputs are obtained by applying a fixed restriction (e.g., decimation or averaging) to fine solutions at each time, and the network learns to invert this artificial map, representative of imperfect measurements yielded by real-world senors. In the latter, we generate coarse trajectories using an actual coarse-grid time integrator and fine trajectories using a fine-grid time integrator, which more closely matches deployment when only a coarse solver is available at inference. Because the coarse solver introduces numerical effects (e.g., truncation, dispersion, and aliasing), the coarse trajectory generally does not coincide with a pointwise downsampling of the fine solution, making

simulation–to–simulation SR a stringent regime for both reconstruction and time forecasting. As a result, SRaFTE is especially valuable when fine solutions are unavailable or too expensive to compute or store beyond those generated for training. Conceptually, SRaFTE also shares the same restrict–evolve–lift paradigm as the equation-free multiscale computation (EFMC) [82, 153], and the SR portion can be understood as a data-driven coarse-grid correction technique within the celebrated multi-grid method [190].

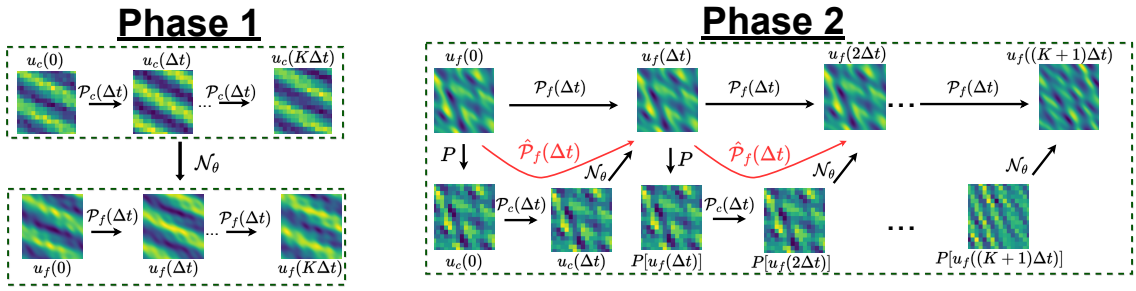


Figure 3.1: **Phase 1 and Phase 2 training in the SRaFTE framework.** In Phase 1, we learn a super-resolution operator that maps the PDE solution at the coarse grid to the fine grid. In Phase 2, we use this learned operator to compose an effective propagator $\hat{\mathcal{P}}_f(\Delta t) := \mathcal{N}_\theta \circ \mathcal{P}_c(\Delta t) \circ P$ that approximates the true fine-grid propagator $\mathcal{P}_f(\Delta t)$.

3.1.1 Main contributions

The main contributions of this work are twofold. First, through systematic numerical experiments on both linear and nonlinear PDEs, we show that the effective fine-grid dynamics propagator obtained by composing a learned super-resolution operator with a coarse-grid numerical integrator enables accurate long-time prediction of time-dependent PDE dynamics. The resulting trajectories exhibit substantially greater stability than those produced by purely data-driven neural operators trained to learn the full solution operator. Second, we provide numerically verifiable local and global approximation error bounds for the effective fine-grid propagator, and further demonstrate that these bounds closely track the observed errors in simulation, thereby explaining the improved numerical stability of SRaFTE in long-time dynamical predictions.

3.1.2 Organization

The remainder of the chapter is organized as follows. Section 3.2 introduces the two-phase SRaFTE framework. Section 3.3 outlines the super-resolution operators used within SRaFTE and the canonical baselines for comparison. Section 3.4 describes the three PDE examples studied in this work and the evaluation metrics used to assess model performance. Section 3.5 presents Phase 1 reconstruction and Phase 2 long-time forecasting results and provides further analysis of the predicted trajectories. In Section 3.6, we conduct a systematic error analysis of SRaFTE. In particular, we derive theoretical local and global approximation error bounds and numerically verify their validity and tightness for a representative nonlinear PDE. Section 3.7 applies SRaFTE to a kinetic electron-dynamics case study via the 1D1V Vlasov–Poisson system, reporting both reconstruction and extrapolation performance together with physics-based diagnostics. Finally, Section 3.8 concludes by summarizing key findings, limitations, and directions for future research. For clarity, all notation used in this work and the associated definitions are summarized in Table A.1.

3.2 SRaFTE: Phase 1 and Phase 2

Let $\Omega \subset \mathbb{R}^D$ be a bounded domain and $X(\Omega)$ be a Banach space of functions on Ω. We consider two spatial discretizations of the bounded domain: $\Omega_f = \{x_i^f\}_{i=1}^{N_f} \subset \Omega$ and $\Omega_c = \{x_i^c\}_{i=1}^{N_c} \subset \Omega$, which correspond to the fine and coarse grids, respectively. For the time-dependent PDE

$$\partial_t u = \mathcal{L} u \quad \text{in } \Omega, \tag{3.1}$$

the operator $\mathcal{L}$ is the infinitesimal generator of a strongly continuous semigroup associated with the PDE. Let $u(\cdot, t) \in X(\Omega)$ denote the solution of the initial value problem (IVP) for (3.1). Its discrete fine-grid and coarse-grid representations are defined as

$$u_f(t) := \Big(u(x_1^f, t), \ldots, u(x_{N_f}^f, t)\Big) \in \mathbb{R}^{N_f},$$
$$u_c(t) := \Big(u(x_1^c, t), \ldots, u(x_{N_c}^c, t)\Big) \in \mathbb{R}^{N_c}.$$

Spatial discretization of $\mathcal{L}$ on the two grids yields approximated semi-discrete evolution equations for $u_f(t)$ and $u_c(t)$:

$$\partial_t u_f(t) \approx \mathcal{L}_f\, u_f(t), \qquad u_f(0) = u_f^0, \tag{3.2}$$

$$\partial_t u_c(t) \approx \mathcal{L}_c\, u_c(t), \qquad u_c(0) = u_c^0, \tag{3.3}$$

where $\mathcal{L}_f, \mathcal{L}_c$ denote the fine-grid and coarse-grid discretization of $\mathcal{L}$ under the chosen boundary conditions, and u_f^0 and u_c^0 are the discrete initial conditions. Consider an equispaced time grid $t_n = n\Delta t$, $0 \leq n \leq N$. Defining $u_f^n := u_f(t_n), u_c^n := u_c(t_n)$, the exact solution can therefore be approximated by the discrete one-step evolution equation:

$$u_f^{n+1} \approx \mathcal{P}_f(\Delta t)\, u_f^n,$$
$$u_c^{n+1} \approx \mathcal{P}_c(\Delta t)\, u_c^n.$$

We write $\mathcal{P}_f(\Delta t) = e^{\Delta t \mathcal{L}_f}, \mathcal{P}_c(\Delta t) = e^{\Delta t \mathcal{L}_c}$, which will be called, respectively, the fine-grid and coarse-grid (one-time) propagators for the discretized dynamics.

SRaFTE can be broken down into two separate learning phases. As shown in Figure 3.1, Phase 1 can be understood as the standard super-resolution task [146, 184, 51, 49, 50, 166, 172]: given K pairs of coarse and fine grid snapshots of the PDE solution at different times $\{(u_c^{(i)}, u_f^{(i)})\}_{i=0}^K$, we learn the neural operator $\mathcal{N}_\theta : u_c \to u_f$ by minimizing the loss function

$$\mathcal{J}_{\mathrm{P1}}(\theta) = \frac{1}{K} \sum_{i=0}^{K} \| u_f^{(i)} - \mathcal{N}_\theta\big[u_c^{(i)}\big] \|_{\ell_1(\mathbb{R}^{N_f})}. \tag{3.4}$$

Here θ is the collection of all trainable parameters of the neural operator $\mathcal{N}_\theta$. $\| \cdot \|_{\ell_1(\mathbb{R}^{N_f})}$ denotes the averaged ℓ_1-norm, which is defined by

$$\|v\|_{\ell_1(\mathbb{R}^{N_f})} := \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} |v_{ij}|.$$

$\{v_{ij}\}_{i=1,j=1}^{H,W}$ are the lattice grid points with $N_f = HW$ and entries. For the entire Phase 1 and Phase 2 learning tasks, the choice of neural operator can be rather flexible and several architectures that we tested will be described later. For future time extrapolation, only knowing the super-resolution neural operator is insufficient. In particular, we note that

$$\Big(\mathcal{N}_\theta \circ \mathcal{P}_c(\Delta t)\Big)[u_c(t)] \neq \Big(\mathcal{P}_f(\Delta t) \circ \mathcal{N}_\theta\Big)[u_c(t)]].$$

Therefore, simply composing $\mathcal{N}_\theta$ with the coarse propagator $\mathcal{P}_c(\Delta t)$ does not yield a good approximation of the fine-grid propagator. The approximation error incurred at each step accumulates over time, causing the predicted fine-grid dynamics to drift rapidly away from the ground truth. Controlling this compounding error is precisely the goal of Phase 2. To this end, as shown in Figure 3.1, we compose the learned super-resolution operator $\mathcal{N}_\theta$ with the coarse one-step propagator $\mathcal{P}_c(\Delta t)$ and a projection operator $P : \mathbb{R}^{N_f} \to \mathbb{R}^{N_c}$, to form an *effective fine-grid propagator*:

$$\hat{\mathcal{P}}_f(\Delta t; \theta) := \mathcal{N}_\theta \circ \mathcal{P}_c(\Delta t) \circ P. \tag{3.5}$$

Here $\mathcal{N}_\theta$ is the super-resolution neural operator trained in Phase 1, while $\mathcal{P}_c(\Delta t)$ and P are fixed operators. Thus in Phase 2, we use the Phase 1 training results as initialization for N_θ and then *fine-tune* the parameters θ via a new loss function.

Specifically, we use paired snapshots of fine-grid dynamics $\{(u_f^{(i)}, u_f^{(i+1)})\}_{i=0}^{K}$ as the training data. Note that this is *not* a new dataset. By (3.5), each fine snapshot is projected to the coarse grid, advanced by one coarse-grid step, and then lifted back to the fine grid using the neural operator. Accordingly, we minimize the discrepancy between the true fine-grid evolution and the effective propagator:

$$\mathcal{J}_{\mathrm{P2}}(\theta) = \frac{1}{K}\sum_{i=0}^{K} \|\mathcal{P}_f(\Delta t)\, u_f^{(i)} - \hat{\mathcal{P}}_f(\Delta t;\theta)\,[u_f^{(i)}]\|_{\ell_1(\mathbb{R}^{N_f})} \tag{3.6}$$

Once trained, the learned effective propagator is used for autoregressive forecasting:

$$\hat{u}_f^{n+1} = \hat{\mathcal{P}}_f(\Delta t;\theta)\,\hat{u}_f^{n}, \qquad n \geq K.$$

After m additional steps (i.e. at endtime $T = m\Delta t$), the fine-grid forecast is

$$\hat{u}_f(K\Delta t + T) = \left[\hat{\mathcal{P}}_f(\Delta t;\theta)\right]^m u_f^K.$$

In principle, the effective fine-grid propagator introduces only minimal computational overhead. The projection operator P is a fixed matrix, and the evaluation of the trained neural operator $\mathcal{N}_\theta$ is inexpensive since as it can be seen as a sequence of function compositions. Thus, the main computational cost at each time step arises from the coarse-grid update $\mathcal{P}_c(\Delta t)$. In the whole process, numerical discretization errors and model errors still accumulate over time.

Nevertheless, because the composed operator incorporates the exact coarse-grid propagator—which preserves the macro-scale features of the underlying PDE—we expect the resulting dynamics to be more stable in future dynamics prediction when compared to pure data-driven autoregressive models commonly used in the literature [102, 30, 37], which lack such physically grounded structure and therefore tend to suffer from rapid error growth during extrapolation.

From a training perspective, training via SRaFTE reduces the computational burden of training the neural network, as it is solely responsible for SR. First, the same neural operator architecture is used in both training stages; only the loss function changes between Phase 1 and Phase 2. Second, no backpropagation through the coarse solver is ever required. Once the coarse solver is determined, it remains fixed, and gradients from backpropagation flow solely through the neural operator $\mathcal{N}_\theta$, whose role is to perform super-resolution. This significantly simplifies training and reduces computational

overhead compared with end-to-end approaches that differentiate through multi-step PDE solvers.

Remark. The effective fine-grid propagator $\mathcal{P}_f(\Delta t)$ in SRaFTE can be interpreted as providing a *Markovian closure* for the fine-grid dynamics. A related approach, SHRED [91], is similar in spirit but instead learns a finite-memory, history-dependent closure $\Phi_h(u_c(t;\tau))$ using a shallow recurrent encoder. In contrast, SRaFTE delegates temporal evolution entirely to the coarse-grid time propagator and restricts learning to a *time-local* super-resolution operator. This perspective suggests a separation of variables: $\mathcal{P}_c(\Delta t)$ governs the (approximately) low-dimensional temporal evolution, while $\mathcal{N}_\theta$ provides the high-dimensional spatial reconstruction. Additional intuition, including linear and nonlinear heuristics, a Mori–Zwanzig interpretation [209, 208, 210], and a discussion of when short-memory assumptions are justified or may fail is provided in Appendix A.1.

3.3 Models and baselines

SRaFTE imposes no *a priori* constraints on the architecture of the neural operator $\mathcal{N}_\theta$. In principle, any neural operator framework developed for super-resolution tasks can be adapted into the two-phase SRaFTE pipeline, which is used first to learn and super-resolve the coarse-scale time dynamics, and then to form an effective one-time propagator for predicting future fine-scale dynamics. In this section, we introduce four super-resolution operators that we use within SRaFTE: U-Net[149], FUnet (our architecture), EDSR[104], and FNO-SR[102] and the baselines for benchmarking the numerical results. Complete model setups and hyperparameter definitions are deferred to Appendix A.3.

U-Net

The U-Net [149] is a classical encoder-decoder architecture widely used in computer vision for image super-resolution. We use U-Net as a classical convolutional based networks within SRaFTE and test its applicability in Phase 2 dynamics predictions.

Fourier U-Net (FUnet)

FUnet, a new architecture introduced in this work, augments the U-Net with a spectral convolutional layer. Specifically, between the encoder and decoder in the bottleneck, we add a spectral convolutional block. The FUnet lifts the features $z \in \mathbb{R}^{C_0}$ from the encoder and computes their discrete Fourier transform: $\widehat{z} = \mathcal{F}[z]$. Subsequently, the lowest R modes are retained and transformed back into the time domain. The decoder then

upsamples to the target resolution in the decoder using pixel-shuffle layers with skip connections from the encoder [163]. The FUnet architecture fuses spectral features with convolutional layers, making it effective both for super-resolution and for extracting scientifically meaningful structure from high-resolution data.

Enhanced Deep Super-Resolution (EDSR)

The image super-resolution EDSR model [104] is adopted here with minimal modifications and placed within SRaFTE. Like the U-Net, the RGB channels used in image processing are replaced by field channels representing the temporal trajectories of the PDE solution. The low-resolution input is first processed by 16 residual blocks, followed by a two-stage pixel-shuffle to achieve the desired resolution increase. A final convolution layer then produces the high-resolution output predictions. EDSR is another convolutional-based model tested within SRaFTE.

Fourier neural operator (FNO)

The FNO was originally proposed in Li et al. [102] to learn a one-step propagator for PDEs. However, as a general neural operator framework, it can also be adapted as a modeling ansatz for a super-resolution operator $\mathcal{N}_\theta$. Specifically, it can be trained to take the numerical solution of a PDE on a coarse grid as input and return the corresponding fine grid solution. This super-resolution variant of FNO (referred to as FNO-SR) can then be used in Phase 2 to form an effective propagator for PDE dynamics. Note that this usage of FNO differs from applying FNO to directly approximate the one-step propagator, which will be referred to as the autoregressive FNO (FNO-AR) in the following discussion.

Upsampling

Upsampling refers to interpolation methods that directly maps the corase-grid solution into fine-grid, typically via interpolation kernels such as nearest neighbor, bilinear, or bicubic. For this work, we adopt bicubic interpolation as a simple baseline for assessing the performance of SRaFTE in fine-scale dynamics reconstruction and prediction. In Phase 1, it maps the coarse grid PDE solution at each timestep to a fine grid solution by evaluating the standard cubic convolution kernel in all spatial directions. In Phase 2, the PDE is first solved on the coarse grid, and the resulting coarse-grained solution at each timestep is then upsampled using the same interpolation. Although simplistic, this baseline is helpful for quantifying how much of SRaFTE's advantage stems from operator learning versus mere smooth spatial upsampling.

Rationale. The selected models and baselines include simple nonparametric interpolation, established super-resolution methods, and autoregressive models for dynamics forecasting. Together, these baselines allow us to compare SRaFTE against the two main alternatives: (i) spatial-only refinement methods and (ii) purely data-driven autoregressive models for temporal dynamics prediction. Within this context, the chosen baselines are sufficient to demonstrate the advantage of SRaFTE's operator-composition framework for dynamics predictions.

3.4 Datasets and evaluation metrics

We now introduce the PDEs we study, briefly describing the generation of the training datasets for each example and the evaluation metrics used to assess the predicted dynamics. More details are provided in Appendices A.4 and A.5.

All systems use periodic boundary conditions, a uniform discretization of the square domain $[0,1] \times [0,1]$ for both the coarse and fine grids, and a timestep $\Delta t = 0.01$. We take the projection operator P to subsample every k-th point from the fine grid (i.e. traditional point sampling/spatial decimation).

2D heat equation Consider the 2D heat equation $\partial_t u = \nu \Delta u + \alpha f$ with parameters $\nu = 10^{-3}$, $\alpha = 10^{-2}$, and forcing term $f(x) = \sin(2\pi x)\sin(2\pi y)$. The equation is solved using a Fourier spectral method.

2D wave equation As an example of an energy-conserving PDE, we consider the linear wave equation $\partial_{tt} u = c^2 \Delta u$ with $c = 0.5$. The equation is solved using an explicit finite-difference leapfrog scheme on both coarse and fine spatial grids to generate the training datasets.

Navier–Stokes equations (NSE) As an example of a nonlinear PDE, we consider the 2D incompressible NSE in vorticity form $\partial_t \omega + u \cdot \nabla \omega = \nu \Delta \omega + f$, $\nabla \cdot u = 0$ with $\nu = 10^{-4}$ and forcing term $f(x) = 0.025\Big[\sin 2\pi(x+y) + \cos 2\pi(x+y)\Big]$. The NSE is solved using a Fourier spectral method. To close the system, we introduce a streamfunction ψ and set $u = \nabla^{\perp}\psi := (\partial_y \psi, -\partial_x \psi)$ with $-\Delta \psi = \omega$.

3.4.1 Training data

For all three examples, we generate 1000 trajectories using random initial conditions detailed in Appendix A.4 to form the training datasets. The neural operator $\mathcal{N}_\theta$ is trained

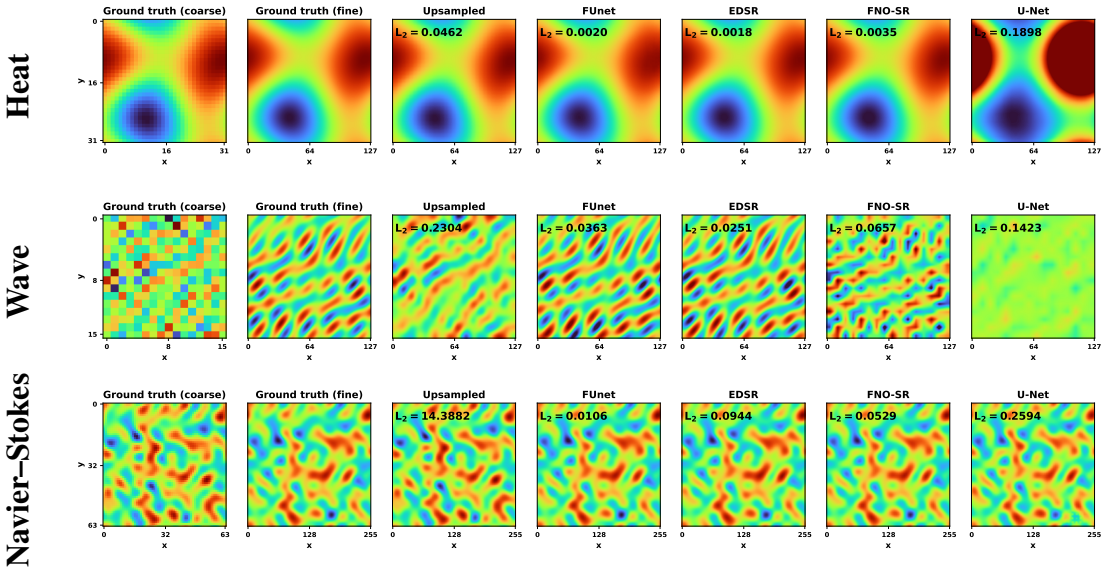


Figure 3.2: **Phase 1 super-resolution results across different PDEs.** For each PDE, we show results for an unseen test trajectory, with representative snapshots from the **(Top)** 2D heat equation, **(Middle)** 2D wave equation, and **(Bottom)** 2D Navier–Stokes equations (NSE). The first column shows the true fine-grid dynamics. The next two columns—labeled "Upsampled" and "FNO-AR"—correspond to the bicubic interpolation and autoregressive FNO baselines, respectively. All subsequent columns display predictions from different SRaFTE models. The inset in each panel reports the relative L_2 reconstruction error.

	Baseline	SRaFTE models			
System	**Upsampled**	**FUnet**	**U-Net**	**EDSR**	**FNO-SR**
Heat equation	5.37×10^{-2}	2.08×10^{-3} $\pm 1.39 \times 10^{-4}$	2.23×10^{-1} $\pm 1.39 \times 10^{-4}$	$\mathbf{6.02 \times 10^{-4}}$ $\pm 8.49 \times 10^{-5}$	3.85×10^{-3} $\pm 4.69 \times 10^{-4}$
Wave equation	2.23×10^{-1}	3.88×10^{-2} $\pm 2.96 \times 10^{-3}$	2.53×10^{-1} $\pm 1.08 \times 10^{-2}$	$\mathbf{3.43 \times 10^{-2}}$ $\pm 2.61 \times 10^{-3}$	7.02×10^{-2} $\pm 2.33 \times 10^{-3}$
Navier–Stokes	1.47×10^{1}	$\mathbf{1.25 \times 10^{-2}}$ $\pm 4.82 \times 10^{-3}$	1.55×10^{-1} $\pm 6.19 \times 10^{-2}$	4.20×10^{-2} $\pm 2.89 \times 10^{-3}$	3.07×10^{-2} $\pm 4.96 \times 10^{-3}$

Table 3.1: **Relative L_2 errors of Phase 1 super-resolution results.** Each error is obtained by averaging over 20 new test trajectories. The shaded column corresponds to the bicubic upsampling baseline, while the remaining columns represent different SRaFTE models. The lowest error in each row is highlighted in bold.

on a paired dataset of coarse-fine dynamics with an 80/20 training/validation split. Specifically, unless otherwise stated, the following coarse-to-fine grid mappings are used: $32 \times 32 \to 128 \times 128$ (heat equation), $16 \times 16 \to 128 \times 128$ (wave equation) and $64 \times 64 \to 256 \times 256$ (NSE). These ratios are chosen after experimenting with a range of downsampling factors, which clearly demonstrate the benefits of super-resolution and future-time extrapolation, particularly in regimes with pronounced multiscale structure.

In Phase 2, the same neural operator $\mathcal{N}_\theta$ is used to construct an effective propagator $\hat{\mathcal{P}}_f(\Delta t; \theta)$, which is retrained using the fine grid data from Phase 1 over a slightly shifted time window $t \in [0, 1 + \Delta t]$ and subsequently used for dynamics extrapolation. For the heat equation, extrapolation is performed until physical time $t = 2$, when the system reaches a steady state. For the wave equation and NSE, the PDEs are evolved up to physical time $t = 10, 7$, respectively, to probe strongly out-of-distribution (OOD) dynamics.

3.4.2 Metrics

To assess the generalizability of the learned super-resolution operator and effective propagator for PDE dynamics prediction, we evaluate the model-predicted dynamics in both phases using a new set of solutions generated from initial conditions randomly sampled from the same distribution. Specifically, we test 20 new trajectories and report the mean and standard deviation across three independent training runs. For Phase 1, we calculate the relative L_2 error across all predictions, while for Phase 2, we compute each of the metrics described below. Full details of each metric can be found in Appendix A.5.

Why multiple metrics? As suggested in Ren et al. [146], simple pixel-wise losses alone can be misleading for scientific data, as lower pixel-wise error does not guarantee physically faithful solutions. Thus, it is necessary to use different metrics to evaluate the

performance of our models, as no single scalar score can capture every facet of fidelity that matters for high-resolution scientific data. Below we characterize the key features of each metric.

- Relative L_2 error ($\mathbf{L_2}$) is *scale-agnostic* but *location-sensitive*: any pixel-wise mismatch is penalized, so large-amplitude errors anywhere in the field dominate. In machine learning contexts, this is one of the most standard error metrics.

- The Structural Similarity Index Measure (**SSIM**) is a perceptual metric designed to assess local, pixel-wise agreement in texture, contrast, and structural patterns. Two fields may be close in the L_2 norm yet differ significantly in visual or structural appearance; SSIM is specifically sensitive to such discrepancies. By definition, SSIM $\in [-1, 1]$, with a value of 1 indicating perfect structural fidelity between the reconstructed and reference fields.

- Spectral mean-squared error (**Spectral MSE**) only measures the *distribution* of energy across Fourier modes. It is particularly sensitive to aliasing, mode transfer distortions, or over-smoothing effects that may not be strongly reflected in either the L_2 error or SSIM

- Pearson correlation (**Corr**) measures the *linear correlations* between the predicted and reference fields. Because it is computed after centering each field, r is *scale-invariant*: two fields that differ only by a global multiplicative factor still achieve $r = 1$. Thus, r complements the other metrics by highlighting how well the spatial patterns vary, irrespective of absolute amplitude.

3.5 Numerical results

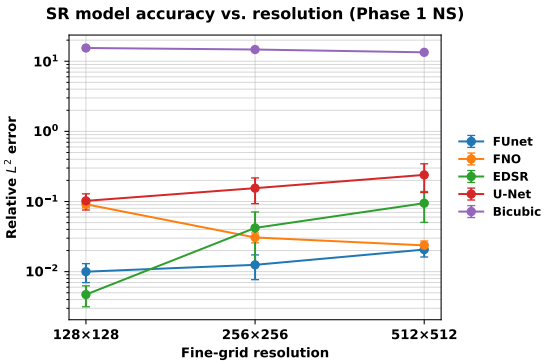
3.5.1 Dynamics super-resolution (Phase 1 test results)

For all the datasets considered, the Phase 1 dynamics reconstruction results and the corresponding evaluation metrics are summarized in Figure 3.2 and Table 3.1. Specifically, Figure 3.2 shows a representative snapshot of the PDE solution at a chosen time ($t = 0.8$, 0.7, and 0.9 for the heat equation, wave equation, and NSE, respectively). Quantitatively, Table 3.1 reports the L_2 errors averaged over all 20 new test trajectories. These results clearly show that the super-resolution models consistently outperform the naive bicubic upsampling baseline by 1–3 orders of magnitude, confirming the utility of the ML-based super-resolution approach for dynamics reconstruction.

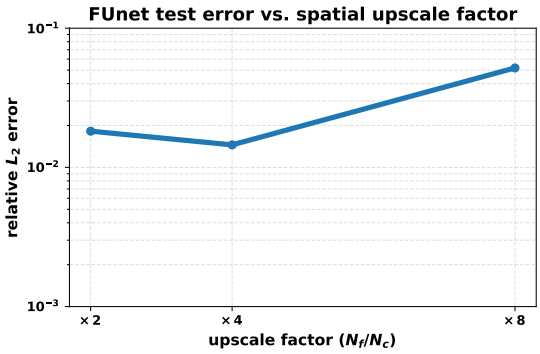
For the three examples considered, the heat equation exhibits the simplest dynamics because it is a linear parabolic PDE with constant coefficients, corresponding to a gradient flow that smooths and decouples Fourier modes via pure exponential decay. As a result, all tested super-resolution models successfully capture the main features of the flow and produce quantitatively more accurate predictions compared to the other two PDEs. The dynamics become more complex for the wave equation, and are further complicated in the Navier–Stokes equation due to nonlinearity. Consequently, as shown in Figure 3.2 and Table 3.1, the discrepancy between super-resolution models and baseline bicubic interpolation increases from the heat equation to the wave equation and then to the NSE. Among all tested SRaFTE models, no significant difference is observed in the Phase 1 results between FUnet, EDSR, and FNO-SR; these three models consistently yield more accurate predictions of the PDE solutions than the purely convolutional-based U-Net model.

To test the scale-robustness of Phase 1 results, we repeat the NS experiment at two additional coarse-fine pair resolutions, $32 \times 32 \to 128 \times 128$ and $128 \times 128 \to 512 \times 512$ for all models and baselines. The resulting resolution sweep, presented in Figure 3.3, tracks the mean and standard deviation of the relative L_2 error for every model as fine grid resolution increases. Across all three resolutions, we see that the FUnet remains the most accurate and shows the slowest error growth. Interestingly, the FNO-SR curve bends downward as we move from the lowest target resolution to the highest, indicating that the model becomes more accurate. This could be because the FNO is inherently biased toward low- and mid-frequency content because each spectral layer retains only a prescribed band of modes [138, 83]. When the fine grid is enlarged while the down-sampling ratio remains fixed, most dynamically important structures now fall inside that retained band, so the proportion of energy that the network must extrapolate beyond its spectral cut-off is reduced.

To further assess the sensitivity of the Phase 1 super-resolution results to input resolution, we evaluate the best-performing model (FUnet) at a fixed fine-grid output resolution of 256×256 while varying the coarse-grid input over $N_c \in \{128, 64, 32\}$ (corresponding to upscale factors $2, 4, 8$). As shown in the right panel of Figure 3.3, the relative L_2 error increases as the coarse resolution decreases, reflecting the expected difficulty of reconstructing fine-scale features from coarser inputs. However, the growth in error remains slow, indicating that the super-resolution performance remains accurate and stable across a broad range of spatial scales. Taken together, these results suggest that the Phase 1 trained FUnet is capable of learning reliable fine-grid reconstructions from coarse-grid inputs across a variety of resolutions.



(a) Accuracy of the super-resolution results for different models with constant downsampling factor of $\times 4$.



(b) Accuracy of the FUnet super-resolution results with different downsamplings factors.

Figure 3.3: **Phase 1 results across different resolutions and scaling factors. (a)** Relative L_2 errors for all models and baselines evaluated at multiple fine-grid resolutions. Each curve is obtained by averaging over 20 unseen trajectories. Models that explicitly exploit spectral structure (FUnet, FNO) achieve the best accuracy, with FUnet exhibiting the slowest growth in error as the fine-grid resolution increases. **(b)** Relative L_2 error of FUnet at a fixed fine-grid resolution of 256×256 while varying the coarse-grid input size $N_c \in \{128, 64, 32\}$ (corresponding to upscale factors $\times 2, \times 4, \times 8$).

		Baselines		SRaFTE models			
System	**Metric**	**Upsampled**	**FNO-AR**	**FUnet**	**EDSR**	**FNO-SR**	**U-Net**
Heat equation	rel. L_2 (↓)	3.66×10^{-2}	$\mathbf{7.07\times10^{-4}}\pm2.9\times10^{-5}$	$1.11\times10^{-3}\pm4.5\times10^{-4}$	$1.47\times10^{-3}\pm6.0\times10^{-4}$	$2.10\times10^{-3}\pm8.6\times10^{-4}$	$2.23\times10^{-3}\pm9.1\times10^{-4}$
	SSIM (↑)	**0.9962**	0.9923 ± 0.0316	0.9950 ± 0.0215	0.9891 ± 0.0447	0.9844 ± 0.0639	0.9806 ± 0.0794
	Spec. MSE (↓)	4.17×10^{-2}	$\mathbf{1.45\times10^{-4}}\pm5.9\times10^{-5}$	$5.89\times10^{-4}\pm2.4\times10^{-5}$	$8.80\times10^{-4}\pm3.6\times10^{-5}$	$2.08\times10^{-3}\pm8.5\times10^{-5}$	$1.55\times10^{-3}\pm6.3\times10^{-4}$
	Corr (↑)	**0.9987**	0.9964 ± 0.0148	0.9982 ± 0.0079	0.9935 ± 0.0266	0.9955 ± 0.0183	0.9773 ± 0.0922
Wave	rel. L_2 (↓)	3.31×10^{-1}	$2.00\times10^{-1}\pm1.1\times10^{-2}$	$2.22\times10^{-1}\pm1.4\times10^{-2}$	$2.52\times10^{-1}\pm1.5\times10^{-3}$	$\mathbf{1.74\times10^{-1}}\pm1.2\times10^{-3}$	$2.06\times10^{-1}\pm1.2\times10^{-1}$
	SSIM (↑)	0.1697	0.2578 ± 0.0340	0.3836 ± 0.0418	0.3667 ± 0.0419	$\mathbf{0.4820}\pm0.0383$	0.3285 ± 0.0371
	Spec. MSE (↓)	1.13×10^{2}	$1.91\times10^{2}\pm1.4\times10^{1}$	$8.75\times10^{1}\pm8.2\times10^{0}$	$8.68\times10^{1}\pm1.1\times10^{1}$	$\mathbf{7.52\times10^{1}}\pm7.1\times10^{0}$	$1.04\times10^{2}\pm2.1\times10^{1}$
	Corr (↑)	0.0479	0.2628 ± 0.0461	0.4072 ± 0.0536	0.3830 ± 0.0553	$\mathbf{0.5393}\pm0.0512$	-0.0764 ± 0.0606
Navier–Stokes	rel. L_2 (↓)	7.92×10^{0}	$1.58\times10^{1}\pm5.4\times10^{-1}$	$\mathbf{9.36\times10^{-2}}\pm7.0\times10^{-3}$	$1.13\times10^{-1}\pm6.5\times10^{-3}$	$1.13\times10^{-1}\pm6.5\times10^{-3}$	$1.68\times10^{-1}\pm7.0\times10^{-2}$
	SSIM (↑)	0.0053	0.453 ± 0.409	$\mathbf{0.9410}\pm0.0518$	0.9210 ± 0.0565	0.9209 ± 0.0573	0.8772 ± 0.0582
	Spec. MSE (↓)	2.76×10^{4}	$5.87\times10^{6}\pm3.0\times10^{5}$	$1.05\times10^{1}\pm1.4\times10^{0}$	$\mathbf{9.11\times10^{0}}\pm1.0\times10^{0}$	$9.19\times10^{0}\pm1.0\times10^{0}$	$1.94\times10^{1}\pm1.4\times10^{0}$
	Corr (↑)	0.3317	NaN	$\mathbf{0.9951}\pm0.0046$	0.9924 ± 0.0058	0.9924 ± 0.0059	0.9887 ± 0.0067

Table 3.2: **Phase 2 future time dynamics prediction across different PDEs.** Each error value is averaged over 20 unseen test trajectories. The shaded baseline columns correspond to bicubic Upsampling and FNO-AR, while the remaining columns show results from various SRaFTE models. The best value in each row is highlighted in bold.

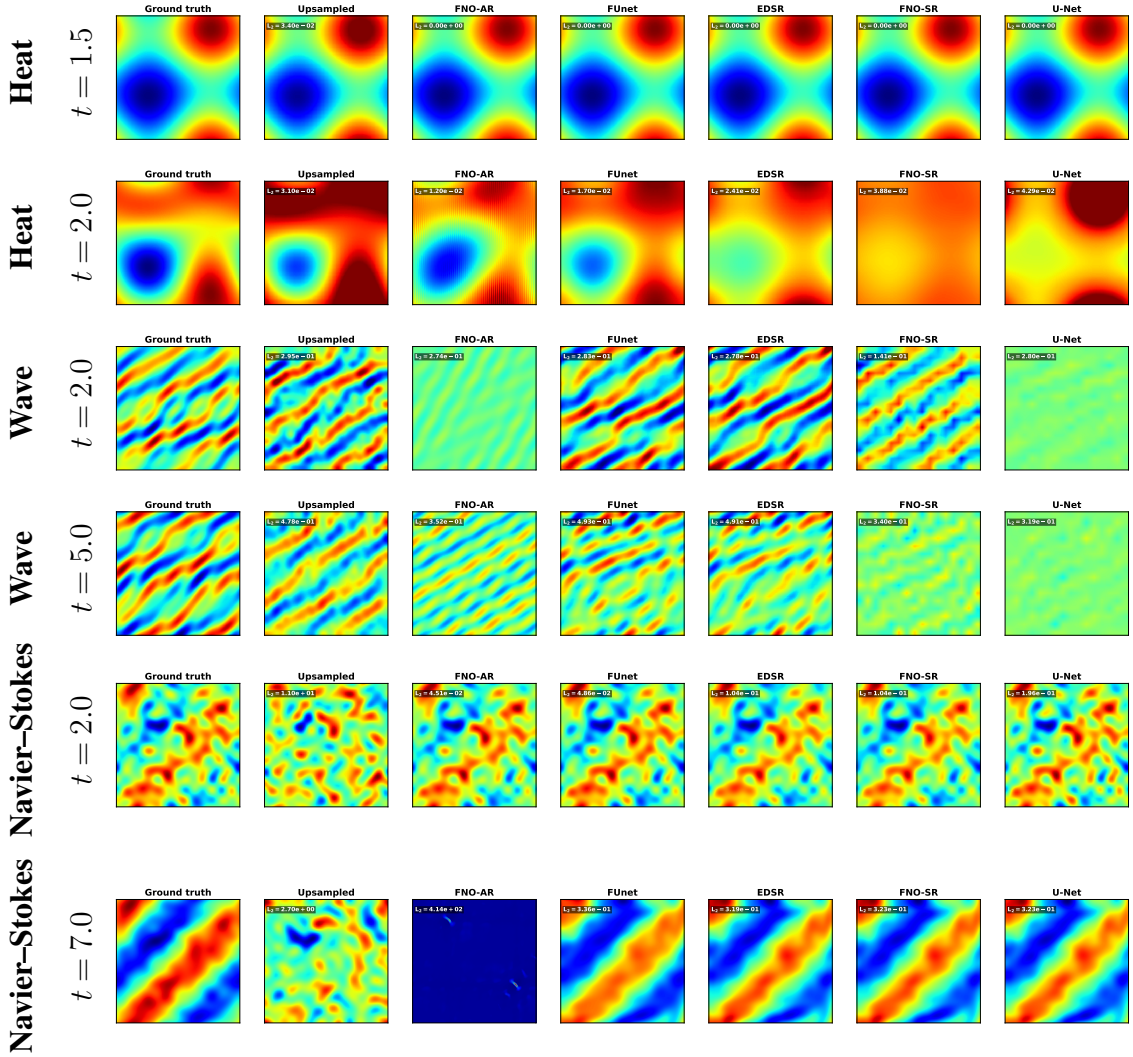
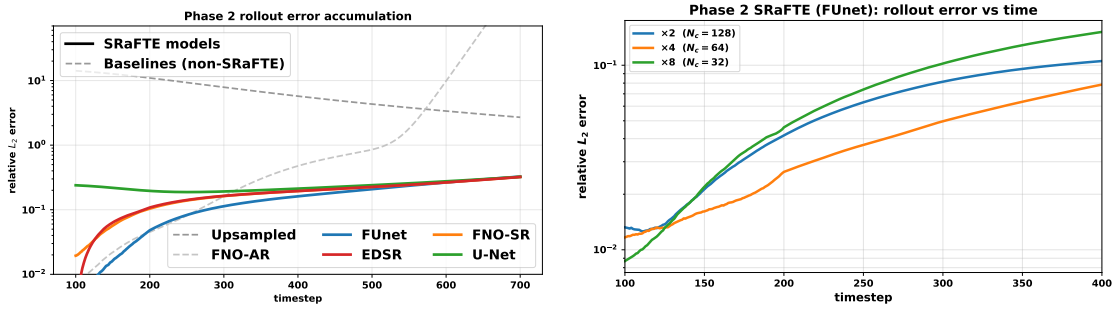


Figure 3.4: **Phase 2 future dynamics prediction results accross different PDEs**. For each PDE, we show results for an unseen test trajectory, with representative snapshots from the **(Top)** 2D heat equation, **(Middle)** 2D wave equation, and **(Bottom)** 2D Navier–Stokes equations (NSE). The first column shows the true fine-grid dynamics. The next two columns—labeled "Upsampled" and "FNO-AR"—correspond to the bicubic interpolation and autoregressive FNO baselines, respectively. All subsequent columns display predictions from different SRaFTE models. The inset in each panel reports the relative L_2 error.



(a) Error accumulation of different models for an new test trajectory at a fixed downsampling scale factor of $4\times$.

(b) FUnet error accumulation by different downsampling scale factor.

Figure 3.5: **Phase 2 error accumulation for NSE. (Left)** Relative L_2 error for a representative unseen test trajectory. At a fixed $4\times$ downsampling factor, the SRaFTE Phase 2 predictions (colored curves) are compared against the baselines (gray). Beyond timestep 400, the autoregressive FNO baseline rapidly diverges, whereas the SRaFTE predictions exhibit substantially slower error growth. **(Right)** Phase 2 rollout error of SRaFTE with the FUnet architecture, illustrating how prediction accuracy scales with the coarse input resolution $N_c \in \{32, 64, 128\}$ while keeping the fine grid fixed at 256×256.

3.5.2 Future dynamics prediction (Phase 2 test results)

For the prediction of future dynamics in each PDE example, we use the effective propagator (3.5) trained in a short time-window $[0, 1]$ to approximate the fine-scale solution. Figure 3.4 presents the dynamics prediction results for a test run with a new random initial condition. Specifically, for the heat, wave, and NS equation, we extrapolate the dynamics from time $t = 1$ to $t = 2, 10, 7$, respectively. Table 3.2 summarizes the averaged statistics over the 20 new test trajectories for each SRaFTE model. Taken together, the results in Table 3.2 and Figure 3.4 lead to the following conclusions:

Numerical advantages for nonlinear PDE From Figure 3.4 and the metric values in Table 3.2, SRaFTE shows clear numerical advantages over baseline methods in predicting the dynamics of the NSE, a nonlinear PDE. This trend is consistent with Phase 1 reconstruction results. The main reason is the rapid accumulation of prediction errors that is especially present in nonlinear systems[30]. As shown in the left panel of Figure 3.5, for an unseen test trajectory, the FNO-AR method quickly accumulates

approximation errors during the dynamics extrapolation phase, leading to large deviations from the true trajectory after only a short time. This effect is also evident in Figure 3.4, where at the final time $t = 7$, the predicted solution blows up.

Across the three PDE examples, the Phase 2 results exhibit a clear and consistent trend: the numerical advantage of SRaFTE increases with the spectral and dynamical complexity of the underlying system. For the heat equation, the fine-scale solution is dominated by low-frequency content, and the autoregressive FNO baseline already performs well; in this regime, SRaFTE models achieve accuracy comparable to FNO-AR. For the linear wave equation, however, the presence of persistent medium-frequency structure makes the prediction task more challenging. Here, coupling the coarse propagator with the learned super-resolution operator in Phase 2 yields a more pronounced improvement over the autoregressive and upsampling baselines. Finally, for the incompressible Navier–Stokes equations, whose nonlinear dynamics contain high-frequency vorticity features, SRaFTE provides the largest gains. The learned super-resolution operator reconstructs fine-scale structures that are absent on the coarse grid, enabling the effective propagator to outperform autoregressive baselines substantially in long-time forecasting.

To test the scale robustness of the Phase 2 results, the right panel of Figure 3.5 repeats the rollout using SRaFTE with the FUnet across three coarse-grid resolutions $N_c \in \{32, 64, 128\}$ while fixing the fine grid at 256. Together with the left panel of Figure 3.5, this indicates that SRaFTE's effective propagator is numerically more accurate and stable than AR baselines for the long-time predictions of the dynamics of nonlinear NSE. Moreover, it is more resilient to moderate changes in spatial sampling of the input, which is crucial for practical usage of the developed approach.

Composition propagator vs. equation-free propagator Comparing the numerical results of FNO-SR and FNO-AR, we can assess the effectiveness of learning an effective propagator via the composition operator (3.5). FNO-AR uses the FNO architecture from its original formulation [102] to learn an equation-free fine propagator; in contrast, FNO-SR learns an effective fine propagator that consists of the composition of a projection, a coarse propagator, and a learned super-resolution operator.

From Figure 3.4 and the metrics in Table 3.2, we observe that FNO-AR is more accurate for the heat equation, whereas FNO-SR performs better for more complex systems such as the non-dissipative wave equation and the nonlinear NSE across all metrics. It is important to note that our extrapolation time window (up to 700 timesteps for NS) are substantially longer than those typically reported in the original FNO literature[102], where rollouts of 10 to 50 steps are common. At these extended time windows, autoregressive error accumulation becomes significantly more pronounced,

especially for the nonlinear case, as each step compounds the approximation error from all previous steps, despite adequate hyperparameter tuning (see Figure 3.5 and analysis of Section 3.6). Furthermore, we use a nonrestrictive spectral cutoff that preserves high-frequency content (Appendix A.4), meaning that the inherent low-frequency bias of FNO-AR becomes especially prevalent at long forecasting time windows [83, 138, 75]. In contrast, SRaFTE's composition structure, where temporal evolution is handled by the coarse solver and the neural operator provides only spatial correction, exhibits the slow-growth additive regime that remains stable over these long time windows. In such cases, the composition operator (3.5) serves as a better ansatz for the fine-scale dynamics solver. This trend is consistent with the numerical performance of the other SRaFTE models discussed above.

SSIM and spectral MSE for perceptual and spectral fidelity Relative L_2 error tends to conflate large-scale shape alignment with fine-scale features, so model predictions that visually differ can have similar L_2 error. Thus, we employ two additional metrics, SpecMSE and SSIM. SSIM compares the structural difference between the predicted flow and the ground truth; hence it is able to track the human-level perceived fidelity of the predictions much more closely. SpecMSE compares the error in the energy spectra of the predicted PDE solutions, a physically relevant metric. Thus, when relative L_2 errors are similar (e.g. for the wave equation), SSIM and SpecMSE offer more discriminative, task-relevant diagnostics to identify the best performing model. This is particularly pronounced in the Phase 2 wave equation results of Table 3.2.

Runtime and storage benefits To further test the generalizability of the learned super-resolutuion operator N_θ for dynamics prediction. For the NSE example, we vary a newly introduced coarse timescale factor, denoted by $s \geq 1$, and use it in the Phase 2 dynamics prediction scheme as: $u^{n+s} \approx \hat{\mathcal{P}}_f(s\Delta t;\theta)\, u^n$. The obtained numerical solution will be benchmarked with the fine-grid one on this coarse time grid. Note that changing s only alters the coarse integrator's step size and thus the temporal spacing from Δt to $s\Delta t$ at test time, and N_θ, P operator will be fixed. Concretely, the fine baseline advances with 600 steps of size Δt, whereas SRaFTE advances with $600/s$ steps of size $s\Delta t$. For a fixed end time of the simulation, we compare the total wall-clock time and the numerical error.

As shown in Figure 3.6, SRaFTE overtakes the fine solver at $s \approx 5$ and yields concrete end-to-end speedups of 1.06× (s=5), 1.27× (s=6), 1.50× (s=7), and 1.71× (s=8). Note that all results here use the FUnet as the neural network model. As expected, the accuracy trade–off increases exponentially with s, providing a transparent compute–fidelity knob. From the test, we see that taking larger coarse-grid time steps Δt can reduce total runtime at acceptable error.

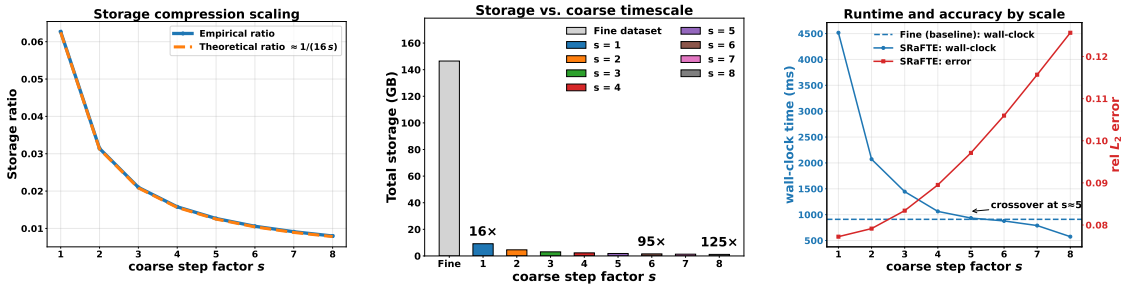


Figure 3.6: **Storage and runtime benefits for NSE.** All results use the FUnet architecture within SRaFTE. **(Left)** Storage compression as a function of the coarse time-scale factor s; the empirical trend closely matches the theoretical $1/(16s)$ scaling implied by $4\times$ spatial downsampling. **(Middle)** Total storage versus s (same trained model and number of trajectories). Values above each bar indicate the corresponding computational savings factor. **(Right)** Efficiency–accuracy tradeoff: total wall-clock runtime (left axis) and time-averaged relative L_2 error (right axis).

To address the dataset level storage tradeoff, we compare storing 1000 full fine trajectories on disk against storing a single SRaFTE trained FUnet plus 1000 coarse trajectories on disk at the same physical time window, shown in the left and center panels of Figure 3.6. Even at $s{=}1$, storing just the weights of the model and the coarse dataset option is already $\approx 16\times$ smaller than storing the fine dataset (9.2 GB vs 145.6 GB). As s increases, total storage decreases further, yielding $\approx 79\times$ at $s{=}5$ and $\approx 125\times$ at $s{=}8$. Practically, this enables saving orders of magnitude less data while retaining the ability to reconstruct fine-resolution fields on demand at evaluation time.

In summary, the numerical experiments demonstrate the effectiveness of the SRaFTE framework in PDE dynamics reconstruction and prediction. When integrated with modern super-resolution architectures, SRaFTE enables the reconstruction of fine-scale numerical solutions of PDEs from the coarse-scale solutions. By design, the composition operator serves as a natural ansatz for constructing an effective propagator to predict future fine-scale dynamics. In the NSE example, we find that SRaFTE substantially outperforms baseline approaches such as FNO-AR and naive upsampling.

3.6 Error Analysis

In this section, we present an error analysis of the SRaFTE-learned effective fine-grid propagator $\hat{\mathcal{P}}_f(\Delta t)$ from Phase 2 for predicting the numerical solutions of PDEs. In standard autoregressive forecasting, one learns a single-step map $\Phi_\theta : \hat{u}^n \mapsto \hat{u}^{n+1}$

directly, so that both spatial refinement and temporal evolution must be inferred by the network. By contrast, our construction uses the compositional operator $\hat{\mathcal{P}}_f(\Delta t;\theta) = \mathcal{N}_\theta \circ \mathcal{P}_c(\Delta t) \circ P$, which explicitly decomposes the task into a *spatial super-resolution step* and a *temporal dynamics propagation step*, with each stage introducing its own approximation error. Accordingly, our analysis focuses on two complementary aspects:

1. **Deterministic (trajectory-wise) error bound.** For the numerical solution of a well-posed initial value problem (IVP) of a PDE with a prescribed initial condition selected *from the training dataset*, we derive explicit bounds for the one-step *local approximation error* and the n-step *global accumulation error*. For each case, we systematically compare the contributions of the *super-resolution approximation error* and the *numerical integration error*, analyzing their relative magnitudes and influence on the overall error accumulation.

2. **Statistical (distributional) generalization bound.** In Appendix A.2, for the numerical solution of a well-posed IVP of a PDE with an initial condition u_0 *randomly sampled* from a distribution $\mathcal{D}$, we derive a bound for the *expected one-step error* in terms of its empirical expectation plus a standard[81, 159] $O(m^{-1/2})$ capacity term, where m denotes the number of training samples. This result quantifies the generalization ability of the SRaFTE-learned effective fine-grid propagator $\hat{\mathcal{P}}_f(\Delta t)$ when applied to new initial conditions $u_0 \sim \mathcal{D}$.

3.6.1 Preliminaries

In the following analysis, we compare the discrete numerical solutions of PDEs generated by two propagators: a sufficiently accurate fine-grid propagator $\mathcal{P}_f(\Delta t) : \mathbb{R}^{N_f} \to \mathbb{R}^{N_f}$ and its SRaFTE-learned approximation $\hat{\mathcal{P}}_f(\Delta t) : \mathbb{R}^{N_f} \to \mathbb{R}^{N_f}$. This formulation allows us to bound the error entirely within a finite-dimensional setting using the ℓ_p norm. For linear PDEs, this connection can be formalized using classical consistency–stability arguments (e.g., Lax–Richtmyer-type results). For nonlinear PDEs, analogous conclusions typically require problem-dependent stability or Lipschitz/contractivity assumptions on the time-Δt solution operator (or on the discrete propagator) to control the propagation of perturbations.

We define the fine-scale spatial discretization of the *exact* PDE solution by $u_f^n = [u(x_j,t_n)]_{j=1}^{N_f} \in \mathbb{R}^{N_f}$, where N_f is the total number of fine grid points. The corresponding coarse-scale representation is written as $u_c^n = [u(x_j,t_n)]_{j=1}^{N_c} \in \mathbb{R}^{N_c}$ with $N_c < N_f$. By definition, we have $Pu_f^n = u_c^n$. Unless otherwise specified, all norms considered below are averaged ℓ_1 norms on $\mathbb{R}^{N_f}$ or $\mathbb{R}^{N_c}$.

To facilitate the error analysis, we introduce the following assumptions and auxiliary results :

(A1) **Coarse propagator regularity.** For fixed $\Delta t = t_{n+1} - t_n$, the coarse-scale numerical integrator $\mathcal{P}_c(\Delta t) : \mathbb{R}^{N_c} \to \mathbb{R}^{N_c}$ is *locally Lipschitz* on a bounded set that contains the relevant coarse-scale states. Specifically, let $\mathcal{D} \subset \mathbb{R}^{N_c}$ denote a bounded set (e.g., a forward-invariant set or a ball containing all coarse trajectories under consideration). Then, there exists a constant $L_c(\mathcal{D}) > 0$ such that for all $w_1, w_2 \in \mathcal{D}$,

$$\|\mathcal{P}_c(\Delta t)w_1 - \mathcal{P}_c(\Delta t)w_2\| \leq L_c(\mathcal{D})\,\|w_1 - w_2\|. \tag{3.7}$$

(A2) **Neural lift regularity.** The *trained and fixed* neural lift operator $\mathrm{N}_\theta : \mathbb{R}^{N_c} \to \mathbb{R}^{N_f}$ is also Lipschitz continuous and satisfies:

$$\left\|\mathcal{N}_\theta[w_1] - \mathcal{N}_\theta[w_2]\right\| \leq L_N\,\|w_1 - w_2\|, \tag{3.8}$$

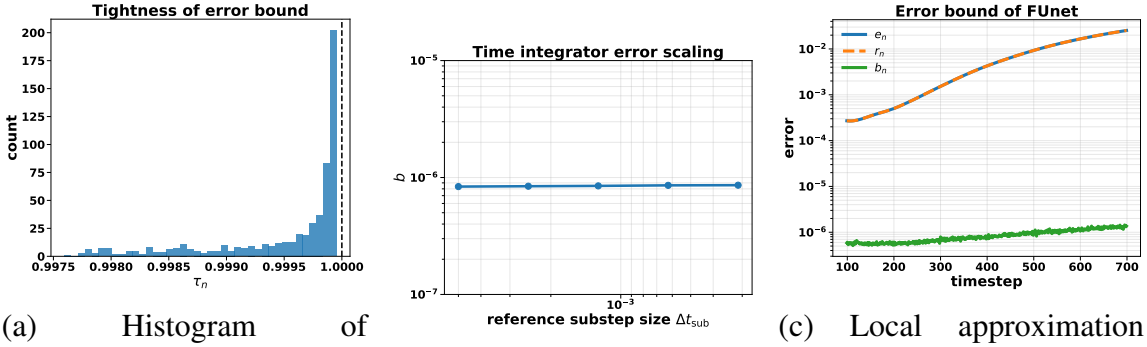
for some $L_N > 0$ and all $w_1, w_2 \in \mathbb{R}^{N_c}$.

(A3) **Error of the coarse numerical integrator.** Let the exact coarse-scale one-step propagator be denoted by $\mathcal{P}_c^{\mathrm{exact}}(\Delta t) : \mathbb{R}^{N_c} \to \mathbb{R}^{N_c}$. The local truncation error associated with a q-th order numerical integrator $\mathcal{P}_c(\Delta t)$ satisfies

$$\left\|\mathcal{P}_c^{\mathrm{exact}}(\Delta t)u_c^n - \mathcal{P}_c(\Delta t)u_c^n\right\| \leq C(\mathcal{D})\,\Delta t^{\,q+1}, \tag{3.9}$$

for all $0 \leq n \leq M$, where $\mathcal{D} \subset \mathbb{R}^{N_c}$ is a bounded set containing the relevant coarse states (e.g., $\{u_c^n\}_{n=0}^M \subset \mathcal{D}$). The constant $C(\mathcal{D})$ depends on the coarse-grid ODE, the numerical scheme, and bounds on the solution/derivatives on $\mathcal{D}$, but is uniform in n and Δt.

For (A1)–(A3), whether the coarse propagator has Lipschitz continuity (A1) can be determined by the ODE system arising from the coarse-grid discretization of the underlying PDE. In particular, as $\Delta t \to 0$, the Lipschitz constant L_c can be well-approximated and calculated from the Lipschitz constant of the corresponding ODE vector field. Conversely, since a standard neural network is a composition of affine transformations and regularized nonlinear activation functions assumption (A2) is also valid. Finally, (A3) is simply the standard local truncation error estimate for finite-difference or numerical integration schemes applied to ODEs.



(a) Histogram of $\tau_n = e_n/(r_n+b_n)$. (b) b vs. substep size Δt_{sub}. (c) Local approximation error.

Figure 3.7: **Local approximation error on a test NSE trajectory. (Left)** Histogram of the ratio $\tau_n = e_n/(r_n + b_n)$ for timesteps $100 \le n \le 700$. **(Middle)** Mean numerical integration error as a function of timestep refinement. **(Right)** Comparison of e_n, r_n, and b_n along a long-time trajectory.

3.6.2 Theoretical local and global error bounds

We first consider the trajectory-wise local approximation error defined by:

$$\varepsilon_{\text{step}}(u_f^n;\theta) := \|\mathcal{P}_f(\Delta t)u_f^n - \hat{\mathcal{P}}_f(\Delta t;\theta)u_f^n\|. \tag{3.10}$$

$$\varepsilon_{\text{step}}(u_f^n;\theta) \le \underbrace{\left\|\left(\mathcal{P}_f(\Delta t) - \mathcal{N}_\theta \circ \mathcal{P}_c^{\text{exact}}(\Delta t)\circ P\right)u_f^n\right\|}_{\mathsf{A}(u_f^n;\theta)} + \underbrace{\underbrace{\left\|\mathcal{N}_\theta\left(\mathcal{P}_c^{\text{exact}}(\Delta t)Pu_f^n\right)\right.}_{\mathsf{B}_1(u_f^n;\theta)} - \underbrace{\left.\mathcal{N}_\theta\left(\mathcal{P}_c(\Delta t)Pu_f^n\right)\right\|}_{\mathsf{B}_2(u_f^n;\theta)}}_{\mathsf{B}(u_f^n;\theta)}. \tag{3.11}$$

Using (A1)–(A3), we can derive a uniform bound for $\mathsf{B}(u_f^n)$ as $\mathsf{B}(u_f^n) \le CL_N\,\Delta t^{q+1}$, where the constant C is independent of u_c^n. Since $\mathcal{P}_c^{\text{exact}}(\Delta t)$ represents the exact one-step propagator of the coarse dynamics, the two terms can be interpreted as follows: $\mathsf{A}(u_f^n;\theta)$ corresponds to the *super-resolution approximation error*, while $\mathsf{B}(u_f^n)$ represents the *numerical integration error*. From the uniform upper bound $\mathsf{B}(u_f^n) \le CL_N\,\Delta t^{q+1}$, we observe that as $\Delta t \to 0$, the contribution of $\mathsf{B}(u_f^n)$ to the total error becomes asymptotically negligible and hence the super-resolution error dominates. It is important to note that this directly justifies the objective function (3.6) we chose for Phase 2 training of the neural network where $\mathsf{A}(u_f^n;\theta)$ is minimized across the training window.

Numerical evaluation of (3.11) Although exact evaluation of the two error terms is generally intractable, they can be well approximated using a highly accurate reference solution that approximates the action of $\mathcal{P}_c^{\mathrm{exact}}(\Delta t)$:

$$z_{\mathrm{ref}}(u_f^n) := \Big[\mathcal{P}_c\Big(\tfrac{\Delta t}{\nu}\Big)\Big]^{\nu} P u_f^n \approx \mathcal{P}_c^{\mathrm{exact}}(\Delta t) P u_f^n,$$

where $\nu \geq 2$ is a temporal refinement factor. At each time step n, the total error and its two contributing components can be tracked by their corresponding numerical approximations:

$$\varepsilon_{\mathrm{step}}(u_f^n;\theta) \approx e_n, \quad \mathsf{A}(u_f^n;\theta) \approx r_n, \quad \mathsf{B}(u_f^n) \approx b_n,$$

where

$$\begin{aligned} e_n &:= \Big\|\mathcal{P}_f(\Delta t)\tilde{u}_f^n - \mathcal{N}_\theta \mathcal{P}_c(\Delta t) P \tilde{u}_f^n\Big\|, \\ r_n &:= \Big\|\mathcal{P}_f(\Delta t)\tilde{u}_f^n - \mathcal{N}_\theta\Big[z_{\mathrm{ref}}(\tilde{u}_f^n)\Big]\Big\|, \\ b_n &:= \Big\|\mathcal{N}_\theta\Big[z_{\mathrm{ref}}(\tilde{u}_f^n)\Big] - \mathcal{N}_\theta\Big[\mathcal{P}_c(\Delta t) P \tilde{u}_f^n\Big]\Big\|. \end{aligned} \tag{3.12}$$

Here $\tilde{u}_f^n := [\mathcal{P}_f(\Delta t)]^n u_f^0 \approx u_f^n$ denotes the approximate fine-scale numerical solution of the PDE. This approximation is valid when $\mathcal{P}_f(\Delta t)$ provides an accurate representation of the exact propagator. All the quantities above are computed using the ℓ_1-norm on $\mathbb{R}^{N_f}$, i.e., evaluated on the fine grid.

Next, we consider the global approximation error of the effective fine-grid propagator $\hat{\mathcal{P}}_f(\Delta t)$. For a neural operator $\Phi_\theta : \hat{u}^n \mapsto \hat{u}^{n+1}$ that learns the discrete-time dynamics of a PDE (e.g., an FNO-type model), classical autoregressive (AR) analysis [117] provides a global error bound in terms of its Lipschitz constant $L_\theta := \mathrm{Lip}(\Phi_\theta)$:

$$\|u_f^n - \hat{u}_f^n\| \lesssim \sum_{j=0}^{n-1} (L_\theta)^{n-1-j}\, e_j. \tag{3.13}$$

For our composite effective map $\hat{\mathcal{P}}_f := \mathcal{N}_\theta \circ \mathcal{P}_c \circ P$, the submultiplicativity of the Lipschitz constant yields

$$\mathrm{Lip}(\hat{\mathcal{P}}_f) \leq L_N\, L_c C_P, \tag{3.14}$$

where L_N and L_c are the Lipschitz constants defined in (3.7)–(3.8), and C_P denotes the induced operator norm of the projection operator P. Combining (3.13)–(3.14), we obtain the following global approximation error bound:

$$\begin{aligned} \|u_f^n - \hat{u}_f^n\| \lesssim \sum_{j=0}^{n-1} (L_N L_c C_P)^{n-1-j} \Big(r_j + b_j\Big) \\ + \mathcal{O}(\Delta t^{p+1} + h_f^l), \end{aligned} \tag{3.15}$$

where u_f^n denotes the exact solution of the PDE evaluated at time $t_n = n\Delta t$ on the fine grid, and $\hat{u}_f^n = [\hat{\mathcal{P}}_f(\Delta t)]^n u_f^0$ denotes the approximate solution obtained via the effective fine-grid propagator $\hat{\mathcal{P}}_f(\Delta t)$. The additional term $\mathcal{O}(\Delta t^{p+1} + h_f^l)$ represents the discretization error associated with a p-th order in time and l-th order in space fine-scale finite difference scheme. For a convergent fine-scale numerical scheme, this error can be made arbitrarily small as $\Delta t, h_f \to 0$. Consequently, the dominant contribution to the global approximation error should also arise from the accumulated super-resolution error $r_j, 1 \leq j \leq n$.

Numerical evaluation of (3.15) Since the one-step error bound can be approximated by r_n and b_n via their definitions in (3.12), numerically approximating the global error bound (3.15) reduces to estimating the Lipschitz constants L_c, L_N, and C_p. By definition, this can be done by randomly sampling inputs x, y within their respective domains and evaluating:

$$\begin{aligned} L_c &= \sup_{x,y \in \mathbb{R}^{N_c}} \frac{\|\mathcal{P}_c(\Delta t)x - \mathcal{P}_c(\Delta t)y\|_{\ell_1(\mathbb{R}^{N_c})}}{\|x - y\|_{\ell_1(\mathbb{R}^{N_c})}}, \\ L_N &= \sup_{x,y \in \mathbb{R}^{N_c}} \frac{\|\mathcal{N}_\theta x - \mathcal{N}_\theta y\|_{\ell_1(\mathbb{R}^{N_f})}}{\|x - y\|_{\ell_1(\mathbb{R}^{N_c})}}, \\ C_p &= \sup_{x \in \mathbb{R}^{N_f}} \frac{\|Px\|_{\ell_1(\mathbb{R}^{N_c})}}{\|x\|_{\ell_1(\mathbb{R}^{N_f})}}. \end{aligned} \tag{3.16}$$

For AR models, the Lipschitz constant $L_\theta := \mathrm{Lip}(\Phi_\theta)$ associated with the single-step map $\Phi_\theta : \hat{u}^n \mapsto \hat{u}^{n+1}$ can be estimated in a similar manner. Combining these two, the error bound analysis can help to explain the observed numerical stability and instability behaviors of these two operator-learning strategies. A central challenge of the autoregressive operator-learning approach is that small one-step approximation errors can rapidly accumulate over time, leading to drift and eventual instability in the predicted trajectories [30, 117]. Such instability is primarily induced by the large Lipschitz constant $L_\theta = \mathrm{Lip}(\Phi_\theta)$ of the learned operator. In contrast, as we shall demonstrate in the next section, the effective Lipschitz constant $L_N L_c C_P$ in the SRaFTE framework is normally much smaller, resulting in a numerically more stable scheme for long-term dynamics predictions.

3.6.3 Numerical evaluation of local approximation errors

We now use the NSE dataset as an illustrative example to numerically evaluate the local error bounds derived in Section 3.6. Specifically, the one-step approximation error

of $\hat{\mathcal{P}}_f(\Delta t)$ is evaluated at *each* time step by comparing the predicted solution $\{\tilde{u}_f^n, \hat{\mathcal{P}}_f(\Delta t)\tilde{u}_f^n\}$ against the reference one $\{\tilde{u}_f^n, \mathcal{P}_f(\Delta t)\tilde{u}_f^n\}$. Recall that we assume the fine-scale solution well-approximates the exact solution, i.e. $\tilde{u}_f^n := [\mathcal{P}_f(\Delta t)]^n u_f^0 \approx u_f^n$. The results presented here correspond to a representative test trajectory and the test for other trajectories yields similar results. Our main findings are summarized as follows:

Bound validity and tightness. To examine the tightness of the local approximation error bound (3.11), we evaluate the ratio between the measured error and the theoretical bound: $\tau_n = e_n/(r_n + b_n)$ over all $100 \leq n \leq 700$ extrapolated time steps, and analyze the corresponding histogram of r_n. As shown in Figure 3.7a, $\tau_n \approx 1$, which confirms that the error decomposition in (3.11) provides a tight bound for the local approximation error.

Negligible numerical integration error. In Figure 3.7b, we plot the mean numerical integration error $b = \frac{1}{N}\sum_n b_n$ as a function of different refinements of the timestep: $\Delta t_{\text{sub}} = \Delta t/n_{\text{sub}}$ for $n_{\text{sub}} \in \{2, 4, 8, 16, 32\}$, obtained using the same coarse-grid solver. Namely, for each n_{sub}, we replace the coarse propagator $\mathcal{P}_c(\Delta t)$ inside $\hat{\mathcal{P}}_f(\Delta t)$ by the substepped composition $[\mathcal{P}_c(\Delta t/n_{\text{sub}})]^{n_{\text{sub}}}$.

If the numerical integration error were a dominant contributor to the one-step approximation error, then refining Δt_{sub} would noticeably decrease both b and, in turn, the mean error $e = \frac{1}{N}\sum_n e_n$. For the selected test trajectory, however, we find that $b \approx 8.49 \times 10^{-7}$, which is four to five orders of magnitude smaller than the observed mean error $e \approx 7.34 \times 10^{-3}$. The growth of e_n, r_n, and b_n in a long extrapolated trajectory is shown more clearly from Figure 3.7c, which further confirms the claim we made by pure theoretical analysis that the numerical integration error introduced through the evaluation of $\mathcal{P}_c(\Delta t)$ in $\hat{\mathcal{P}}_f(\Delta t)$ is negligible compared to the super-resolution approximation error. We also repeated this experiment using Richardson extrapolation on the same coarse integrator and obtained nearly identical values of b, leading to the same conclusion that time-discretization error is negligible compared to the super-resolution approximation error.

It is important to emphasize that this observation does not imply (i) that using the coarse grid would resolve the underlying physics, or (ii) that all numerical artifacts are negligible. Rather, it establishes that, for the chosen coarse discretization, the numerical integration error is insignificant relative to the super-resolution approximation error. The observed difference between the upsampled baseline and the SRaFTE results for the NS equation can thus be attributed to the fact that the learned super-resolution operator $\mathcal{N}_\theta$ more accurately recovers the missing subgrid information and mitigates the projection–lifting mismatch.

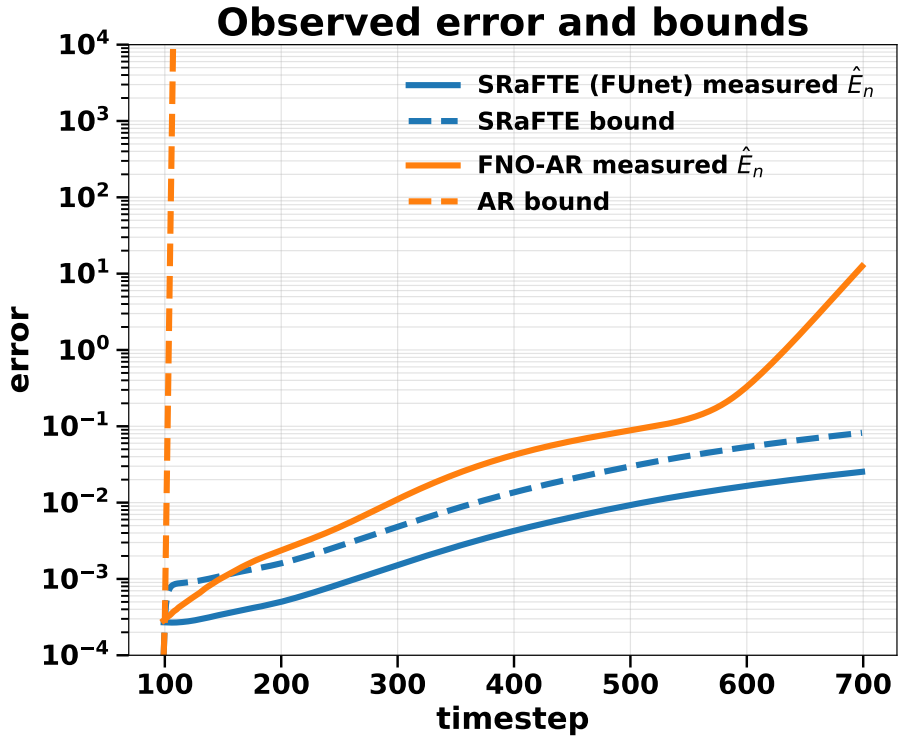


Figure 3.8: Measured global approximation errors and their respective bounds for SRaFTE and the AR baseline.

3.6.4 Numerical evaluation of global approximation errors

The global approximation error can be evaluated by comparing the *entire* predicted trajectory $\{\hat{u}_f^n\}_{n=0}^N$ with the corresponding reference trajectory $\{\tilde{u}_f^n\}_{n=0}^N$, where $\hat{u}_f^n = [\hat{\mathcal{P}}_f(\Delta t)]^n u_f^0$ and $\tilde{u}_f^n = [\mathcal{P}_f(\Delta t)]^n u_f^0$. Define

$$\hat{E}_n = \|\tilde{u}_f^n - \hat{u}_f^n\|$$

as the global approximation error at step n. The error bound (3.15) then implies

$$\hat{E}_n \lesssim \sum_{j=0}^{n-1} (L_N L_c C_P)^{n-1-j} e_j, \tag{3.17}$$

where $L_N L_c C_P$ represents the Lipschitz constant of the surrogate propagation operator, which can be approximated by Monte Carlo sampling and evaluating (3.16). Similarly, for a neural operator $\Phi_\theta : \hat{u}^n \mapsto \hat{u}^{n+1}$ with empirical Lipschitz constant $L_\theta := \mathrm{Lip}(\Phi_\theta)$, the corresponding global approximation error bound (3.13) can be expressed as

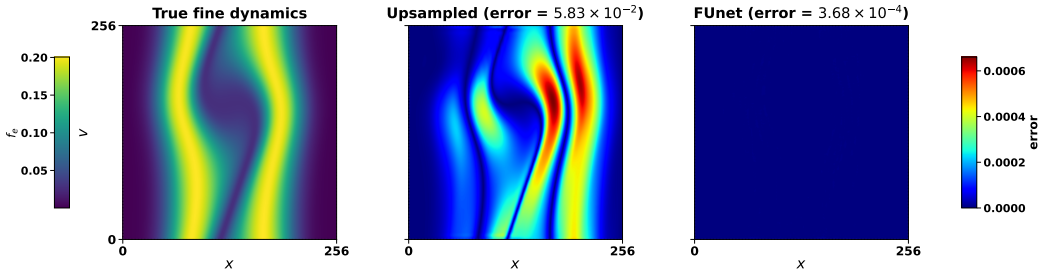
$$\hat{E}_n \lesssim \sum_{j=0}^{n-1} L_\theta^{n-1-j} e_j, \tag{3.18}$$

which can be evaluated numerically in the same way. For a test trajectory of the NS equation, Figure 3.8 compares the observed global approximation error with the theoretical error bounds (3.17)–(3.18) for the SRaFTE-learned effective fine propagator $\hat{\mathcal{P}}_f(\Delta t)$ and the FNO-AR-learned solution operator Φ_θ, respectively. We find that the composite Lipschitz constant $\Lambda = L_N L_c C_P \approx 0.68$, so that the geometric weights Λ^{n-1-j} effectively attenuate the contribution of the one-step approximation errors. As a result, the theoretical bound (3.17) closely tracks the actual global error $\hat{E}_n$. In contrast, for FNO-AR, the empirical Lipschitz constant is $L_\theta \approx 3.11$, which causes the corresponding error bound (3.18) to grow rapidly, reflecting the fast error accumulation observed in the FNO-AR predictions of the PDE solution.

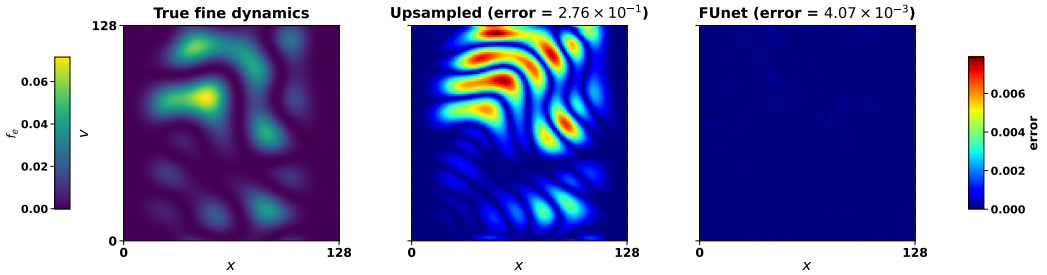
Hence, for standard autoregressive operator-learning methods such as FNO-AR, the global approximation error grows significantly faster than that of the SRaFTE-learned dynamical propagator. This behavior is consistently observed in our numerical simulations, and the corresponding global error bound analysis provides a coherent theoretical explanation for the observed discrepancy in error growth rates.

3.7 Electron dynamics application

Electron-scale phenomena are notoriously costly to simulate because often times the physics of interest only occur on very fine space-time grids [23, 100, 197, 195, 194, 188,



(a) Two-stream instability electron species f_e at $t = 22.5$.



(b) Randomized electrons f_e at $t = 0.5$.

Figure 3.9: **Phase 1 test set results for 1D1V Vlasov–Poisson (two-stream instability and randomized datasets) electron distribution f_e at $t = 22.5$ and $t = 0.5$, respectively**. We switch to error heatmaps to make spatially localized discrepancies explicit and to better separate methods whose predicted fields appear visually similar. The titles indicate the relative L_2 errors.

189]. Under uniform refinement in d dimensions, halving the mesh width increases the number of degrees of freedom by approximately 2^d, and typically tightens stability constraints on the time step. Fast electron time scales and sub-Debye spatial structure further amplify this cost, making long-horizon studies and parameter sweeps impractical for conventional solvers. These regimes therefore provide a stringent and practically motivated test suite for SRaFTE; accordingly, we next consider the classic Vlasov–Poisson system as a representative model for electron dynamics within a plasma.

3.7.1 Vlasov–Poisson equation

The Vlasov–Poisson (VP) equation governs the dynamics of collisionless plasmas. The term *collisionless* means that the particle's motion is primarily governed by electromagnetic fields or gravitational forces, rather than collisions with other particles; consequently, the particle's behavior is not significantly influenced by direct, physical impacts with other particles. As such, equations such as the VP equation are of great interest as many applications in nuclear fusion rely on accurate quantification of plasma dynamics [198, 9, 55].

In full, the collisionless Vlasov–Poisson equation under the effect of electric fields is given as,

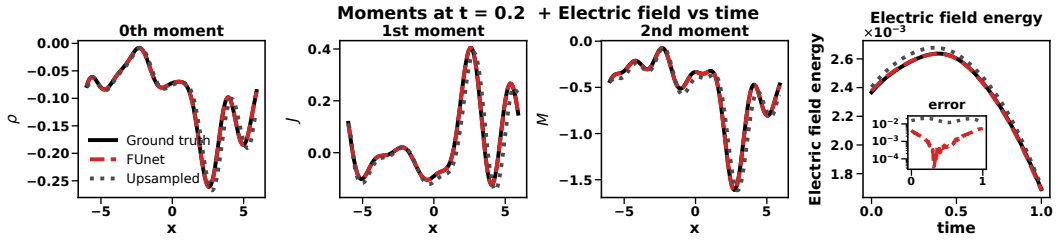
$$\frac{\partial f_s}{\partial t} + \mathbf{v}_s \cdot \nabla_{\mathbf{x}} f_s + \frac{q_s}{m_s} \mathbf{E} \cdot \nabla_{\mathbf{v}} f_s = 0. \tag{3.19}$$

Here, f_s represents the time-dependent distribution function of a particle species s, i.e. electrons or ions, with real-space coordinates $\mathbf{x}$ and velocity-space components $\mathbf{v}$. The species mass is denoted by m_s and the species charge is denoted by q_s. The prescribed electric field $\mathbf{E}$ acts in spatial coordinates $\mathbf{x}$ and is computed via Poisson's equation:

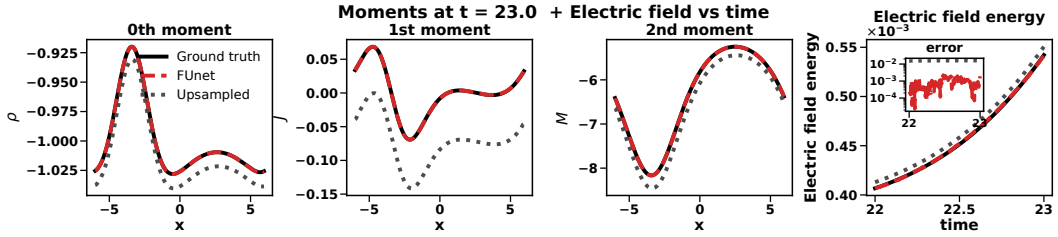
$$\nabla^2 \phi = -\frac{1}{\epsilon_0} \sum_s q_s \int d\mathbf{v}\, f_s(\mathbf{x}, \mathbf{v}, t), \tag{3.20}$$

$$\mathbf{E} = -\nabla \phi. \tag{3.21}$$

Here, ϕ is an electric field and ϵ_0 is known as the *permittivity of free space*; this is a fundamental constant that quantifies how much electric field can fill a vacuum, relating the electric charge density to the resulting electric field [55]. In its entirety, (3.19) is a (6 + 1)-dimensional PDE, as the real and velocity space coordinates are typically 3-dimensional. The nonlinearity in (3.19) arises from the definition of the electric field. In the following results, we assume that the ions are fixed in time (see Appendix A.7), and we focus on modeling the electron species. Furthermore, for this chapter, we will be focused on a 1D1V (2 + 1)-dimensional version of the problem, aligning this setting with our previous results.



(a) Randomized dataset observables and electric field energy density at $t = 0.21$.



(b) Two-stream instability observables and electric field energy density at $t = 23.0$.

Figure 3.10: Prediction of moments at a fixed point in time for Phase 1 and error in electric field energy density for both the two-stream instability and randomized datasets.

3.7.2 Physics-based metrics

For the 1D1V Vlasov–Poisson test cases, we also track system-specific, physics-based diagnostics. In addition to the metrics reported in Section A.5, we employ the following physics based metrics for the 1D1V VP system: the negative-energy penalty, the entropy error, and the electric field error. First, we note that the VP electron distribution function is a probability density, and must remain non-negative. Any predicted negative values are nonphysical, and thus will degrade moment accuracy. The *negative-mass penalty* is thus given by,

$$\mathcal{P}(t) = \iint \left| \min\left(\tilde{f}_e(x,v,t), 0\right) \right| dv\, dx.$$

$\mathcal{P}(t)$ measures the total negative mass in phase space. Ideally, this quantity is as close to 0 as possible as it is unphysical to have a negative mass. Second, for the *entropy error*, we note that for an exact solution, the entropy

$$S(t) = \iint f_e(x,v,t) \ln f_e(x,v,t)\, \mathrm{d}v\, \mathrm{d}x$$

should be conserved in time. Given a prediction $\tilde{f}_e$ we define

$$\tilde{S}(t) = \iint \tilde{f}_e \ln \tilde{f}_e \,\mathrm{d}v \,\mathrm{d}x, \qquad \mathcal{E}_S(t) = \frac{\left|\tilde{S}(t) - S(t)\right|}{\left|S(t)\right|}.$$

A small $\mathcal{E}_S$ signals that the prediction has not introduced spurious velocity-space dissipation. Finally, we can measure the *electric-field error*. From the predicted charge density, $\tilde{\rho}(x,t) = 1 - \int \tilde{f}_e \,\mathrm{d}v$, we solve $\partial_{xx}\tilde{\phi} = \tilde{\rho}$ and set $\tilde{E}(x,t) = -\partial_x \tilde{\phi}$. Comparing with the ground-truth field $E = -\partial_x \phi$ gives

$$\mathcal{E}_E(t) = \frac{\left\|\tilde{E} - E\right\|_2}{\|E\|_2}.$$

A low $\mathcal{E}_E$ indicates that the predicted charge distribution produces an electrostatic force consistent with Gauss' law, ensuring accurate particle acceleration and energy exchange.

Furthermore, for Vlasov–Poisson systems, physical observables in the form of velocity moments are sometimes the quantities of interest for plasma physicists [2, 55]. Many physically relevant quantities pertaining to plasmas are encoded within the velocity moments, including the charge density, the current density, and the kinetic energy density. These moments and other higher-order moments provide a compact macroscopic fingerprint of the plasma's underlying dynamics in the form of quantities that physicists can measure. Concretely, given a species distribution $f_s(\mathbf{x}, \mathbf{v}, t)$, we define its velocity moments by

$$M_s^{(k)}(\mathbf{x}, t) := \int_{\mathbb{R}^{d_v}} \mathbf{v}^{\otimes k} \, f_s(\mathbf{x}, \mathbf{v}, t) \, d\mathbf{v}, \qquad k = 0, 1, 2, \ldots, \tag{3.22}$$

where d_v is the velocity-space dimension and $\mathbf{v}^{\otimes k}$ denotes the k-fold outer product. In particular, the zeroth moment is the number density,

$$\rho_s(\mathbf{x}, t) := \int_{\mathbb{R}^{d_v}} f_s(\mathbf{x}, \mathbf{v}, t) \, d\mathbf{v} = M_s^{(0)}(\mathbf{x}, t), \tag{3.23}$$

and the charge density appearing in Poisson's equation is the species-weighted sum

$$\rho(\mathbf{x}, t) := \sum_s q_s \, \rho_s(\mathbf{x}, t) = \sum_s q_s \int_{\mathbb{R}^{d_v}} f_s(\mathbf{x}, \mathbf{v}, t) \, d\mathbf{v}. \tag{3.24}$$

The first moment gives the current density,

$$\mathbf{J}(\mathbf{x}, t) := \sum_s q_s \int_{\mathbb{R}^{d_v}} \mathbf{v} \, f_s(\mathbf{x}, \mathbf{v}, t) \, d\mathbf{v} = \sum_s q_s \, M_s^{(1)}(\mathbf{x}, t), \tag{3.25}$$

and the second moment we report in our results is

$$\mathbf{M}_s(\mathbf{x}, t) := \int_{\mathbb{R}^{d_v}} \mathbf{v} \otimes \mathbf{v} \, f_s(\mathbf{x}, \mathbf{v}, t) \, d\mathbf{v} = M_s^{(2)}(\mathbf{x}, t). \tag{3.26}$$

	Two-stream		Randomized ($m_x = m_v = 4$)	
	Bicubic	**FUnet**	**Bicubic**	**FUnet**
(a) Flow diagnostics				
f_e	5.799×10^{-2}	$\mathbf{4.378 \times 10^{-4}} \pm 1.7 \times 10^{-5}$	2.468×10^{-1}	$\mathbf{6.160 \times 10^{-3}} \pm 4.4 \times 10^{-4}$
(b) Moment errors				
ρ	1.242×10^{-2}	$\mathbf{7.466 \times 10^{-5}} \pm 4.3 \times 10^{-6}$	1.229×10^{-1}	$\mathbf{3.439 \times 10^{-3}} \pm 3.1 \times 10^{-4}$
J	1.746×10^{0}	$\mathbf{4.547 \times 10^{-3}} \pm 3.3 \times 10^{-4}$	2.895×10^{-1}	$\mathbf{9.912 \times 10^{-3}} \pm 1.3 \times 10^{-3}$
M_2	3.758×10^{-2}	$\mathbf{1.458 \times 10^{-4}} \pm 8.8 \times 10^{-6}$	1.547×10^{-1}	$\mathbf{7.575 \times 10^{-3}} \pm 9.1 \times 10^{-4}$

Table 3.3: Phase 1 test set results for the Vlasov–Poisson datasets. Errors are reported in relative L_2. Columns group the two-stream instability and randomized distribution ($m_x = m_v = 4$) test cases. Bold indicates the best *mean* within each dataset for a given metric. Bicubic interpolation is reported as means only since it is deterministic.

3.7.3 Phase 1 test results

We begin by demonstrating the test set results of Phase 1 for the 1D1V VP system for both the two-stream instability and randomized electron datasets. For testing each different system, we draw new initial conditions from the same random distributions that were used for training, and propagate both the coarse-grain and fine-grain solvers until the same fixed final physical time. For the randomized dataset, we use $T = 1$ and for the two-stream instability dataset, we use $T = 24$. For the two-stream instability data set, we learn a $64 \times 64 \to 256 \times 256$ mapping, and for the randomized dataset, we learn a $32 \times 32 \to 128 \times 128$ mapping. Finally, since FUnet consistently performed best across the preceding case studies, we report the results below only for this choice of neural operator.

Figures 3.9 and 3.10 alongside Table 3.3 summarize the Phase 1 test set reconstruction performance for the 1D1V Vlasov–Poisson benchmarks. Figure 3.9 shows representative fine-grid snapshots of the electron distribution $f_e(x, v, t)$ for an unseen two-stream trajectory (at $t = 22.5$) and an unseen randomized trajectory (at $t = 0.5$), comparing bicubic upsampling against the FUnet prediction. Here, we observe error heatmaps because they localize where the super-resolution mapping succeeds or fails in phase space. In contrast to the previous cases, we found it more difficult to discern differences purely visually based on the fields themselves. In both test cases, the bicubic upsampling baseline produces an overly smooth reconstruction with systematic

magnitude mismatch relative to the ground truth, while the FUnet more faithfully matches both the amplitude and the localized fine-scale variability. This is consistent with the previous results from Section 3.5.

Quantitatively, Table 3.3 reports mean relative L_2 errors over 20 new test trajectories for both the field-level reconstruction of f_e and the derived velocity-moment observables ρ, J, and M. For the two-stream case, the relative L_2 error in f_e drops from 5.79×10^{-2} (bicubic) to 4.37×10^{-4} (FUnet), and the moment errors improve by multiple orders of magnitude (e.g., ρ from 1.24×10^{-2} to 7.46×10^{-5}). For the randomized dataset, FUnet similarly reduces the reconstruction error in f_e from 2.46×10^{-1} to 6.16×10^{-3}, with consistent reductions in ρ, J, and M. Figure 3.10 further illustrates that these gains persist at the observable level: the predicted moment profiles closely track the ground truth at fixed times, and the resulting electric-field energy error remains small, indicating that the learned lift improves not only pixel-wise agreement, but also physically relevant aggregates driven by phase-space structure. Finally, for the sake of clarity, we add Figures A.5 and A.6, which shows the evolution in time of the ground truth dynamics, baseline, and our Phase 1 super-resolution predictions.

An interesting question one can investigate is the transferability of high frequency training to a lower frequency test set with no fine-tuning. This zero-shot generalization phenomenon has also been noticed in a variety of different scientific contexts, including weather and fluid turbulence modeling [171, 37, 30]. In Appendix A.8, we explore this phenomenon in our electron dynamics context. The main findings indicate the ability to transfer high frequency training to zero-shot (i.e. no fine-tuning) low frequency generalization, but not vice-versa (Figure A.2 and Table A.2). Furthermore, mathematical and empirical analysis of the model weights demonstrate an interpretable analysis that shows that high frequency training yields a more expressive model, thus is capable of generating fine-scale features that low frequency training misses (Figure A.3 and Figure A.4).

In summary, for the Vlasov–Poisson benchmarks, the Phase 1 learned lift N_θ reconstructs the fine-grid phase-space state from coarse inputs with high fidelity; in representative test cases, the resulting field-level agreement is roughly two orders of magnitude better than bicubic upsampling, and this improvement propagates to the derived observables ρ, J, and M, where we observe up to three orders of magnitude in terms of error improvement.

3.7.4 Phase 2 test results

These Phase 1 results motivate the Phase 2 evaluation, where we compose the coarse Vlasov–Poisson update with the learned lift to obtain an effective fine-grid propagator and

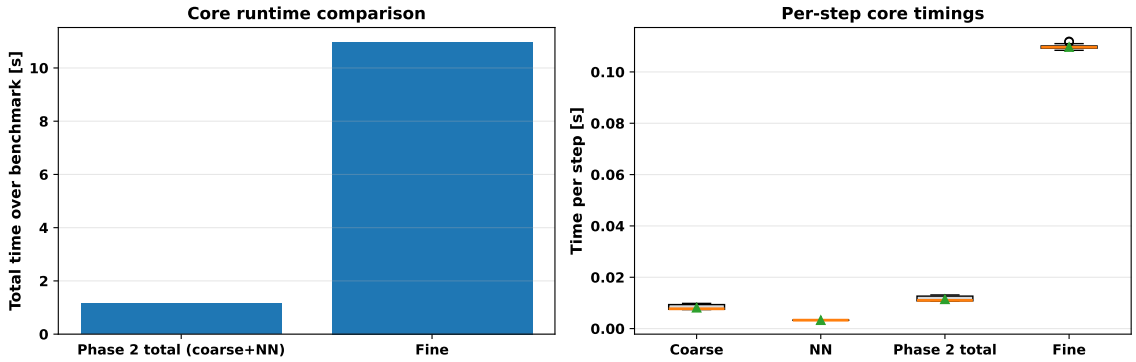


Figure 3.11: **Core runtime comparison and per-step timings.** We compare one fine-grid Vlasov–Poisson step on a 256×256 phase-space grid against our Phase 2 extrapolation, which consists of the projection into the coarse grid, the time stepping on the coarse grid, and a single neural super-resolution query to the fine-resolution. **Left**: total time accumulated over a fixed number of benchmark steps. **Right**: decomposed per-step distributions. The Phase 2 pathway achieves a consistently lower runtime envelope on our node, with approximately $11\times$ reduction.

assess both efficiency and rollout accuracy. For the two-stream instability, we roll out to physical time $T = 28$, corresponding to 400 time steps beyond the training window (a $4\times$ extrapolation relative to the training horizon); for the randomized dataset, we roll out to $T = 3$, corresponding to 200 steps beyond the training window (a $2\times$ extrapolation).

We first quantify the core runtime advantage of the Phase 2 pipeline for the VP datasets in Figure 3.11 by comparing a single fine-grid step on a 256×256 phase-space mesh against one Phase 2 step consisting of a 64×64 coarse physics update followed by a single neural upscaling query on GPU. Both the accumulated time benchmark and the per-step distributions show that the Phase 2 pathway maintains a consistently lower runtime envelope than the fine solver, with the coarse update dominating the Phase 2 per-step cost and the neural lift contributing only a modest overhead. Note that, in contrast to the preceding PDE solvers, we do not additionally coarse-grain the time step here. Due to the greater complexity of the Vlasov–Poisson update in time, substantial wall-clock savings are already realized under the solver's natural time stepping. Finally, we note that the fine-scale dataset requires approximately 80 GB of storage, whereas the coarse-scale dataset together with the neural network weights requires only about 4 GB, yielding roughly a $20\times$ reduction in storage cost.

Turning to accuracy, Figure 3.12 compares representative snapshots in the future time extrapolation for the two-stream and randomized datasets. In both cases, advancing on

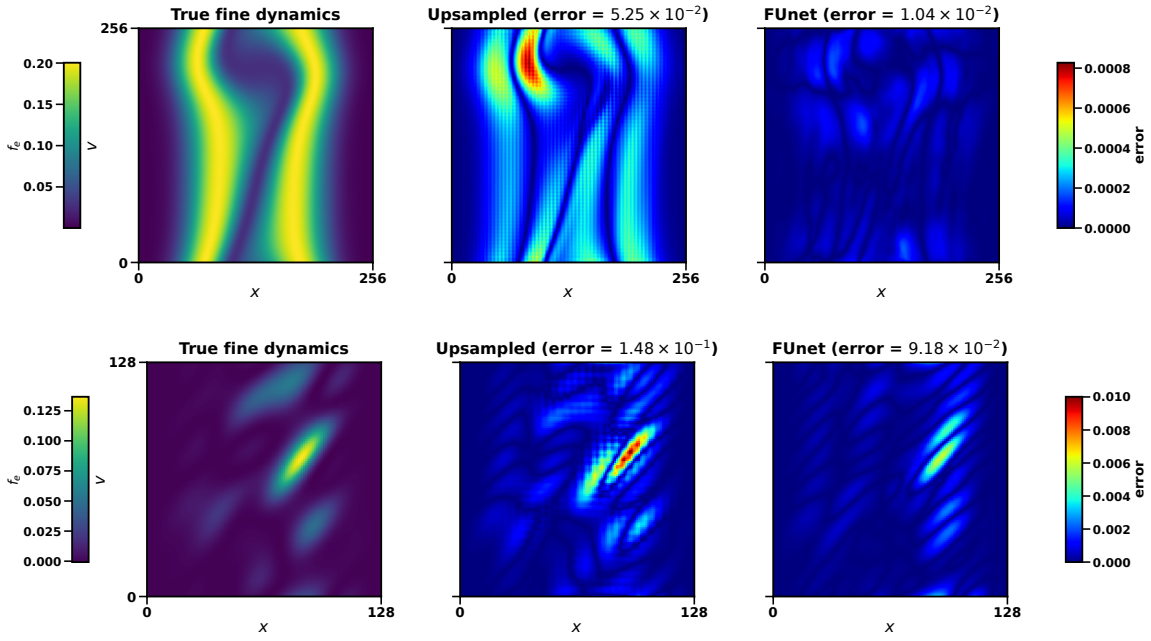


Figure 3.12: Phase 2 long-horizon Vlasov–Poisson rollouts for the two-stream and randomized datasets: ground truth fine-grid f_e, bicubic coarse-to-fine lifting, and SRaFTE (coarse step + learned lift), with relative L_2 errors reported in each panel.

Metric	Baseline: Upsampled	SRaFTE model: FUnet
Two-stream		
(a) Flow / physics diagnostics		
rel. L_2 (↓)	5.18×10^{-2}	$\mathbf{7.86\times10^{-3}}\pm3.70\times10^{-4}$
SSIM (↑)	9.85×10^{-1}	$\mathbf{9.99\times10^{-1}}\pm5.60\times10^{-5}$
Spect. MSE (↓)	2.07	$\mathbf{5.89\times10^{-2}}\pm4.50\times10^{-3}$
PSNR (↑)	31.00	$\mathbf{49.10}\pm6.60$
Neg-mass pen. (↓)	**0.00**	$1.95\times10^{-5}\pm3.00\times10^{-7}$
Entropy err. (↓)	7.09×10^{-4}	$\mathbf{5.66\times10^{-4}}\pm6.70\times10^{-5}$
E-field err. (↓)	8.08×10^{-2}	$\mathbf{4.12\times10^{-2}}\pm1.40\times10^{-3}$
(b) Velocity-moment errors (relative L_2)		
ρ	3.38×10^{-3}	$\mathbf{1.08\times10^{-3}}\pm5.60\times10^{-5}$
J	1.56	$\mathbf{7.27\times10^{-2}}\pm2.80\times10^{-3}$
M	8.75×10^{-3}	$\mathbf{5.31\times10^{-3}}\pm2.50\times10^{-4}$
Randomized ($m_x = m_v = 4$)		
(a) Flow / physics diagnostics		
rel. L_2 (↓)	1.66×10^{-1}	$\mathbf{9.32\times10^{-2}}\pm6.30\times10^{-3}$
SSIM (↑)	9.17×10^{-1}	$\mathbf{9.63\times10^{-1}}\pm3.10\times10^{-3}$
Spect. MSE (↓)	1.83×10^{-1}	$\mathbf{7.94\times10^{-2}}\pm7.70\times10^{-3}$
PSNR (↑)	3.28×10^{1}	$\mathbf{4.15\times10^{1}}\pm1.00\times10^{0}$
Neg-mass pen. (↓)	5.99×10^{-1}	$\mathbf{7.89\times10^{-2}}\pm4.10\times10^{-3}$
Entropy err. (↓)	7.68×10^{-2}	$\mathbf{6.60\times10^{-3}}\pm6.40\times10^{-4}$
E-field err. (↓)	$\mathbf{8.84\times10^{-3}}$	$8.77\times10^{-2}\pm7.10\times10^{-3}$
(b) Velocity-moment errors (relative L_2)		
ρ	6.06×10^{-2}	$\mathbf{4.33\times10^{-2}}\pm2.50\times10^{-3}$
J	1.34×10^{-1}	$\mathbf{8.52\times10^{-2}}\pm5.20\times10^{-3}$
M	1.14×10^{-1}	$\mathbf{7.99\times10^{-2}}\pm4.70\times10^{-3}$

Table 3.4: **Vlasov–Poisson Phase 2 rollout metrics (top: two-stream; bottom: randomized distribution).** Block (a) lists flow/physics diagnostics; block (b) reports relative L_2 errors for velocity moments. Within each case, the best mean per row is highlighted in bold.

the coarse grid and lifting via bicubic interpolation yields noticeably larger phase-space mismatch compared to Phase 2 extrapolation with the FUnet. In the shown examples, the reported errors decrease from 5.25×10^{-2} to 1.04×10^{-2} (two-stream) and from 1.48×10^{-1} to 9.18×10^{-2} (randomized). To make the implications for physically meaningful aggregates explicit, Figure 3.13 plots the 0th, 1st, and 2nd moments ρ, J, and M. Here, we see that the learned Phase 2 rollouts track the ground-truth profiles more closely than bicubic lifting, most prominently for the current density J.

Finally, Table 3.4 summarizes test-set rollout metrics. For the two-stream case, the

learned Phase 2 propagator improves field-level errors (e.g., rel. L_2 from 5.18×10^{-2} to 7.86×10^{-3}, spectral MSE from 2.07 to 5.89×10^{-2}, PSNR from 31.0 to 49.1) and reduces physics/observable discrepancies (e.g., J error from 1.56 to 7.27×10^{-2}, and electric-field error from 8.08×10^{-2} to 4.12×10^{-2}).

For the randomized case, FUnet again improves most metrics, including relative L_2 error (1.66×10^{-1} vs. 9.32×10^{-2}), spectral MSE (1.83×10^{-1} vs. 7.94×10^{-2}), and entropy error (7.68×10^{-2} vs. 6.60×10^{-3}), while the relative electric-field metric is the main exception, favoring bicubic lifting (8.84×10^{-3} vs. 8.77×10^{-2}). We suspect that the electric field diagnostic in this dataset can favor a smoother lift because E is obtained from ρ via Poisson's equation, which emphasizes large-scale (low-k) charge modes and effectively filters small-scale phase-space errors. Consequently, bicubic interpolation may achieve a smaller E-error by damping high-frequency density fluctuations, even when its phase-space and moment-level matches are worse.

Finally, for the sake of clarity, we add Figures A.7– A.10 which show the evolution in time of the ground truth dynamics, baseline, and our Phase 2 extrapolation predictions for the electron species of both datasets and its relevant moments.

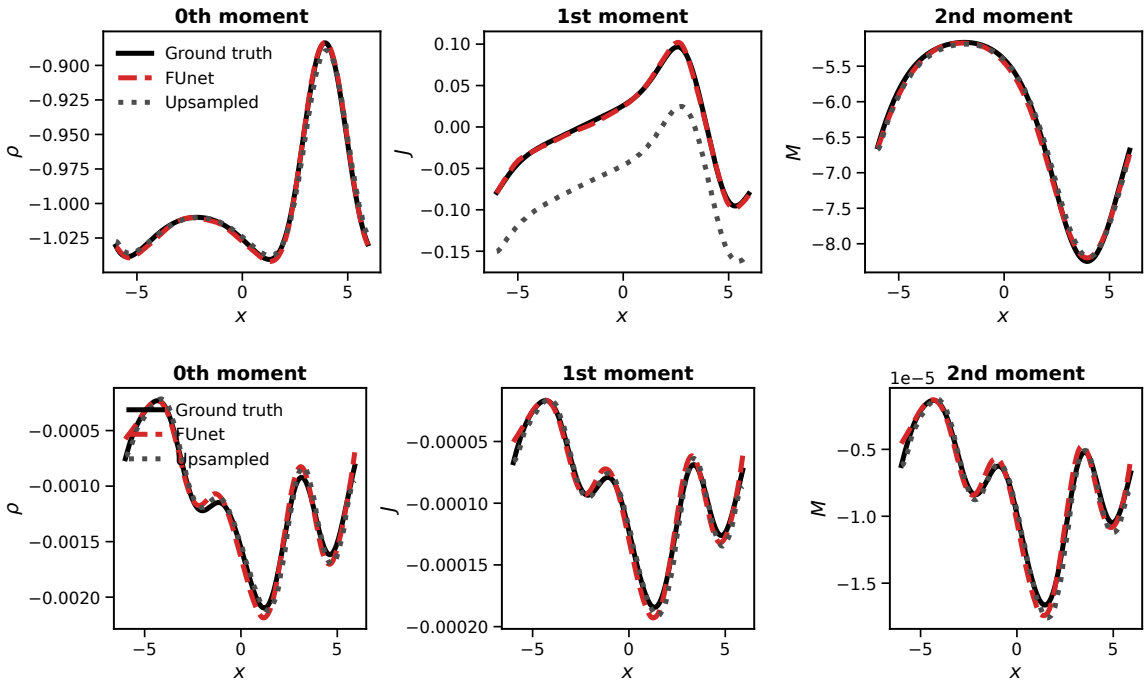


Figure 3.13: **Phase 2 Vlasov–Poisson rollout diagnostics for the two-stream instability *(top)* and randomized *(bottom)* datasets**: velocity moments ρ, J, and M comparing ground truth, bicubic upsampling, and Phase 2 SRaFTE predictions.

3.8 Conclusion

In this chapter, we introduced SRaFTE, a two-phase learning framework that couples a super-resolution neural operator with a coarse-grid numerical integrator to approximate the fine-scale dynamics of PDEs. Across different PDEs studied in the work, SRaFTE successfully reconstructs high-resolution fine-grid solutions from coarse inputs (Phase 1) and delivers stable, accurate long-time forecasts (Phase 2) of the underlying dynamics. We also provided a systematic error analysis of the SRaFTE-learned effective fine-grid propagator, demonstrating that the derived local and global approximation error bounds closely follow the observed numerical errors in simulations of a representative nonlinear PDE.

While this work focuses on PDE dynamics on regular geometries with uniform discretizations, future extensions may incorporate adaptive meshes, irregular spatial domains, or more complex boundary conditions. An additional direction is to extend SRaFTE to *non-autonomous* settings, such as PDEs with time-dependent forcing, in which the induced propagator becomes explicitly time-dependent and the learned model must account for temporal variability beyond the state dependence alone. Finally, it would be interesting to complement the empirical results with learning-theoretic guarantees for the propagator approximation task, for example by studying whether an empirical Rademacher complexity estimate for the SRaFTE hypothesis class exhibits favorable scaling with the number of training trajectories or snapshots (e.g., polynomial or logarithmic dependence on m). Overall, our results highlight SRaFTE as a practical and flexible framework for pairing learned neural operators with existing numerical solvers, offering a scalable path toward high-resolution forecasting of spatiotemporal systems.

Chapter 4

From noisy observables to accurate ground state energies: a quantum-classical signal subspace approach with denoising

4.1 Introduction

4.1.1 Overview

Ground state estimation is a fundamental challenge across numerous branches of science and engineering, including chemistry, physics, and materials science. Understanding the ground state of a many-body quantum system provides critical insights into the system's properties. While classical diagonalization of the exact many-body Hamiltonian scales exponentially in the system size, quantum computing has the potential to solve such eigenvalue problems with theoretically guaranteed polynomial scaling [27, 96, 144, 106]. These guarantees, however, rest on the core assumption of perfect fault-tolerance and error correction. As such capabilities remain out of reach for practical quantum hardware, there is a growing need to leverage current devices, despite their individual limitations. Hybrid quantum-classical approaches have been developed to exploit the capabilities of current noisy intermediate-scale quantum (NISQ) platforms. The idea of hybrid algorithms is to delegate to NISQ devices routines that best use their inherent strengths, for example representing and simulating many-body wave functions,

[0]The following chapter is based on published work in *APL Quantum* [6].

and then to employ classical computers to post-process the results measured on the NISQ device [54, 132, 88, 43, 161, 119, 177, 44, 42, 162].

One hybrid method for ground state estimation is observable dynamic mode decomposition (ODMD) [161]. ODMD assumes that one has already produced a time series of scalar observables via successive time evolution of a reference state. The observables can be computed cheaply on quantum hardware, without computing full time-evolved states in an exponentially large Hilbert space. Next, on classical hardware, one applies a variant of dynamic mode decomposition (DMD) [154, 92] to post-process the time series of observables. In the idealized, noise-free setting, the observables evolve as a linear combination of complex exponentials, where each term corresponds to an eigenfrequency. ODMD serves as a system identification and signal processing tool: when applied to a time series consisting of complex sinusoids, ODMD recovers the leading eigenfrequency component that contains the ground state energy (GSE).

Numerically, ODMD involves solving one least squares problem and one eigenvalue problem; neither of these steps are computationally intensive. ODMD is exponentially convergent in the noise-free setting and resilient to perturbative noise [161]. As noise increases, this resilience weakens, as reflected in ODMD's reduced stability and reliance on hyperparameter tuning within a delicate, often unforgiving range. Both theory and empirical findings motivate the search for a *denoising* procedure that one can apply to the noisy time series of quantum observables to bring it closer to its ideal, noise-free counterpart.

The primary contribution of this chapter is to augment ODMD with Fourier-based denoising procedures. In regimes where ODMD struggles to converge—especially under suboptimal hyperparameter choices—*these denoising procedures substantially improve GSE estimation accuracy without requiring any extra quantum resources*. One of our key ideas is to apply *hard thresholding* to the observable time series' discrete Fourier transform (DFT). Low-amplitude frequency components are interpreted as spurious noise modes and truncated away [66, 115]. The other key idea is *stacking*, which uses ODMD with multiple denoised realizations of the noisy time series. Here each realization corresponds to denoising with a different threshold in Fourier domain. As we show through multiple empirical tests, for noisy data where ODMD exhibits slow or no convergence, the combination of hard thresholding and stacking enables fast convergence to the true GSE. As an aside, we also demonstrate that an enhancement called *zero-padding* [66] mitigates grid resolution issues that arise when estimating the GSE using Fourier-based methods.

4.1.2 Related Work

To our knowledge, this work presents the first use of Fourier-based denoising via frequency domain truncation in quantum algorithms for GSE estimation. Fourier-based approaches, in the context of quantum signal processing, have been applied to various computational tasks [105, 124, 1]. Recent work has incorporated Fourier-based noise suppression via frequency domain windowing into a hybrid workflow to recover accurate single-particle spectra from noisy quantum correlator data [183]. Wang *et al.* use Hamming-window filtering and cross-spectral principal-component analysis to re-weight, but not discard frequency components. In contrast, our method applies a hard threshold to remove low-amplitude Fourier modes outright, producing denoised trajectories that serve as the signal subspace for subsequent DMD.

There exist numerous classical techniques [170] that estimate the frequency set $\{\omega_j\}$ of a time series consisting of a linear combination of complex exponentials $\{\exp(i\omega_j t)\}$. Classical Fourier-based estimation methods (such as periodograms) have a frequency resolution proportional to $1/K$, where K is the length of the time series. DMD is an example of a super-resolution method that goes beyond the classical limit; other examples, which share structural features with DMD, include MUltiple SIgnal Classification (MUSIC) [156, 103], Prony's method [137, 158], and Estimation of Signal Parameters via Rotational Invariant Techniques (ESPRIT) [151, 99, 41]. In the idealized noise-free regime, these algorithms' computational complexity is proportional to the signal's sparsity in the frequency domain, not the number of samples K. This ability to estimate spectra accurately from only a limited number of samples is a hallmark of super-resolution techniques.

A variety of hybrid quantum-classical algorithms [88, 162, 43, 161, 42] have been recently developed and analyzed for eigenenergy estimation, offering alternatives that bypass stringent fault-tolerance requirements of standard quantum phase estimation (QPE) [87]. In contrast to leading hybrid methods that solve generalized eigenvalue [88, 162] or non-linear optimization [43, 44] problems to recover eigenenergies, ODMD casts the task as a matrix least-squares problem. This formulation counteracts ill-conditioning often associated with nonorthogonal bases in generalized eigenvalue problems, while also sidestepping the computational overhead incurred by non-linear optimization.

We remark that some methods [43, 44, 42] further require that the initial state $|\phi_0\rangle$ must have dominant overlap ($p_0 > 0.5$) with the ground state for desirable convergence guarantees. This non-trivial requirement may demand substantial quantum resources at the reference state preparation stage [107]. By comparison, ODMD is provably convergent without such dominance assumptions. In addition to eigenenergy estimation, ODMD canonically incorporates eigenstate recovery, noise source estimation,

super-resolved spectral analysis, and better conditioning via a least-squares framework; to our knowledge, no other hybrid quantum-classical or Fourier-based method delivers all four of these advantages simultaneously.

Overall, this work establishes a robust hybrid algorithm that leverages frequency domain denoising with a quantum signal subspace approach to improve many-body ground state energy estimation from noisy quantum data. Throughout this chapter, we focus our numerical demonstrations on molecular datasets, which underpin electronic structure calculations in quantum chemistry and pose unique challenges for classical algorithms. Unlike familiar spin Hamiltonians where interactions are local and amenable to classical compression techniques such as tensor networks, molecular Hamiltonians involve long-range Coulomb interactions and system-specific orbitals that extend polynomially with the total count of electrons. These features can lead to highly entangled and/or strongly correlated many-body states, implying a steep computational cost and making molecular systems compelling targets for hybrid quantum algorithms [95, 118, 94, 5].

For the specific molecules examined in this work, we focus on GSE estimation in the low ground state overlap, intermediate-to-high noise regime. Combining Fourier denoising with ODMD is especially effective here, enabling desired accuracy and more optimal use of NISQ platforms with a significantly reduced computational budget.

4.1.3 Organization

In Section 4.2, we review necessary background including the measurement of quantum observables via the Hadamard test, Fourier representation of time series, and basic models of noise effects perturbing the observables. In Section 4.3, we describe our methodology. We first overview the ODMD approach for solving the ground state problem. Next, we introduce our novel Fourier denoising ODMD (FDODMD) approach, which integrates frequency domain thresholding with signal stacking. We also describe a time domain zero-padding procedure to enhance Fourier-based ground state energy estimation. In Section 4.4, we detail how we generate data and how we test the convergence of algorithms. We discuss in Section 4.5 a motivating example that involves data with both shot noise and depolarizing error. In Section 4.6, we summarize NISQ-era practicalities required for our methods, including deployment guidance for hardware and high-level resource considerations. In Section 4.7, we compare FDODMD with baseline ODMD and classical (including zero-padded) Fourier-based spectral estimation for the chromium dimer. In Section 4.8, we derive error bounds that help to explain the performance improvements of FDODMD. Section 4.9 summarizes our findings and discusses their broader implications, along with directions for future work. In

Section B.1, we estimate the quantum resources needed for ODMD/FDODMD to compute the GSE of LiH. In Sections B.2 to B.5, we discuss optimal shot allocation, Hamiltonian scaling, results for LiH, and formal proofs of bounds, respectively.

4.2 Background

4.2.1 Acquiring data from a quantum circuit

Consider the expectation values

$$\langle U \rangle = \langle \phi_0 | U | \phi_0 \rangle, \tag{4.1}$$

where $|\phi_0\rangle$ is a fixed reference quantum state and U is a unitary operator. Given a many-body Hamiltonian H, we can efficiently simulate the one-parameter family of time evolution operators $U(t) = e^{-iHt}$ on quantum hardware due to its native unitarity. These real-time expectation values $\langle U(t_k) \rangle$ evaluated at discrete times t_k form the basis of several recently proposed hybrid algorithms for energy estimation [161, 160, 43, 88].

To access the overlap (4.1) on a quantum device, we employ the Hadamard test [106, 35], a standard subroutine for measuring the expectation value of a general unitary operator. Since (4.1) is complex-valued, two Hadamard test circuits are executed, one for the real part and one for the imaginary part. The Hadamard test is visualized in the upper left panel (light gray) of Fig. 4.1. The circuits use a single ancilla and, for each shot, yield a binary outcome corresponding to the measurement of the ancilla in the $|0\rangle$ or $|1\rangle$ basis state, respectively. Averaging over many shots generates an unbiased estimate for the desired overlap. For our purposes, we run the classically simulated Hadamard test to obtain time series of quantum observables.

Note. Throughout this chapter, we use the term *observable* in an operational sense — the quantity obtained by a single-ancilla measurement from a quantum computer — rather than the stricter sense of a Hermitian operator.

The approaches introduced in this chapter require as input a molecular Hamiltonian H and a reference state $|\phi_0\rangle$ with overlap probability $p_0 = |\langle \psi_0 | \phi_0 \rangle|^2$, where $|\psi_0\rangle$ is the true ground state. Suppose we have data on an equispaced temporal grid with $t_k = k\Delta t$ for $0 \leq k \leq K$. Given the above, we measure the real and/or imaginary parts of the *signal*:

$$s(t_k) = \langle U(t_k) \rangle = \langle \phi_0 | e^{-iHk\Delta t} | \phi_0 \rangle. \tag{4.2}$$

In practice, however, measurements performed on current quantum hardware are subject to various sources of error [39, 148, 72, 53]. Two of the most prevalent sources are gate error, which results from imperfect quantum operations, and shot noise, which reflects the

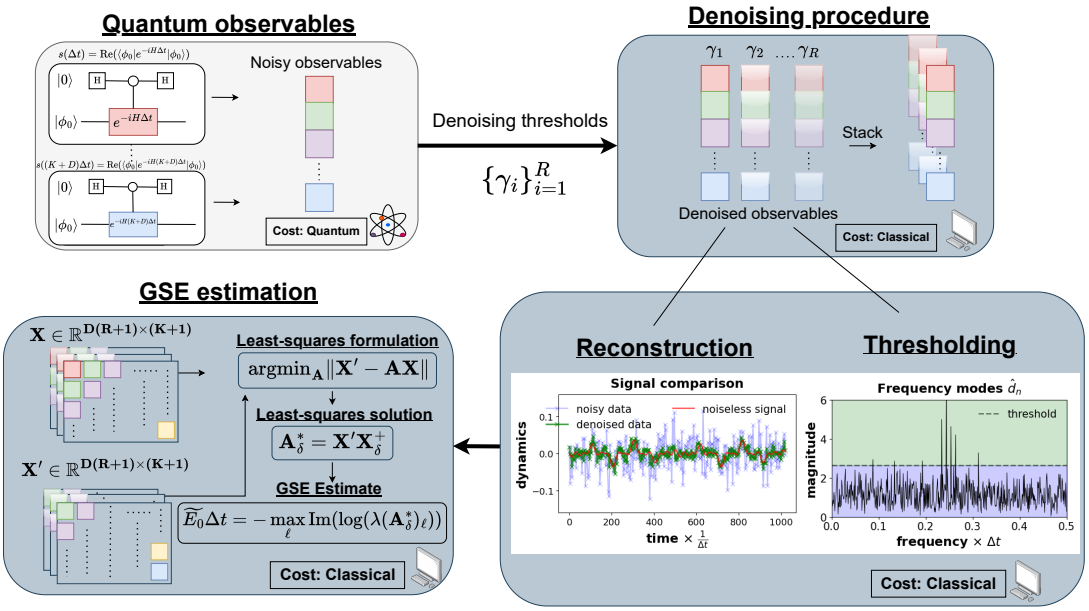


Figure 4.1: Total workflow of FDODMD as outlined in Section 4.3. **(Top left)** The Hadamard test is used to obtain noisy measurements of the data (4.3). **(Top right)** The denoising procedure is then applied by taking R different truncation factors in (4.17). The data is then stacked together according to (4.18). **(Bottom right)** Here we illustrate the thresholding and denoised reconstruction procedures described in Section 4.3.2. **(Bottom left)** The stacked data is used to estimate the GSE, as described in Section 4.3.3.

statistical fluctuations of a finite sampling of shots. Since precisely modeling gate error is highly device-specific and technically challenging [53, 148], we focus primarily on shot noise and examine a simplified global model of gate error.

As a result, when implementing the Hadamard test on NISQ hardware, we measure a noisy version of the real-time overlap (4.2), which we write schematically as the *data*:

$$d(t_k) = s(t_k) + \xi(t_k), \tag{4.3}$$

where $\xi(t_k)$ represents the noise. We emphasize that the input data for all our methods are either the noisy observables (4.3), or a denoised version of them as described in Section 4.3.

4.2.2 Modeling noise effects

Here we discuss how to model the quantum noise $\xi(t_k)$ when constructing (4.3). In principle, $s(t_k) = \langle\phi_0|e^{-iHt_k}|\phi_0\rangle$ is a complex-valued scalar. For simplicity, we focus on

the real part; the imaginary part can be treated analogously.

Since the Hadamard test yields outcomes ± 1, the readout of shot i at time t_k is a Bernoulli random variable $b_i(t_k)$ that takes values $+1$ and -1 with respective probabilities π_{t_k} and $1-\pi_{t_k}$. To ensure that the Bernoulli estimator is unbiased, we require $\mathbb{E}[b_i(t_k)] = \mathrm{Re}\, s(t_k)$, which implies $\pi_{t_k} = (1+\mathrm{Re}\, s(t_k))/2$.

We assume that the random variables $\{b_i(t_k)\}$ are independent and identically distributed. At time t_k, the noisy observable $\mathrm{Re}\, d(t_k)$ is the empirical mean over a total of n_{t_k} shots:

$$\mathrm{Re}\, d(t_k) = \frac{1}{n_{t_k}} \sum_{i=1}^{n_{t_k}} b_i(t_k). \tag{4.4}$$

One can now check that $\mathbb{E}[\mathrm{Re}\, d(t_k)] = \mathrm{Re}\, s(t_k)$ and that

$$\mathrm{Var}(\mathrm{Re}\, d(t_k)) = 4\pi_{t_k}(1-\pi_{t_k})/n_{t_k}, \tag{4.5}$$

which vanishes as $n_{t_k} \to \infty$. This confirms that in the infinite-shot limit, $\mathrm{Re}\, d(t_k)$ converges to $\mathrm{Re}\, s(t_k)$, as desired.

To emulate shot noise in our simulations, we compute the exact value of (4.2) and then add zero-mean Gaussian noise,

$$\mathrm{Re}\, d(t_k) = \mathrm{Re}\langle \phi_0 | e^{-iHt_k} | \phi_0 \rangle + \xi_k, \quad \xi_k \sim \mathcal{N}(0, \epsilon), \tag{4.6}$$

where the standard deviation ϵ can be chosen to match $\sqrt{\mathrm{Var}(\mathrm{Re}\, d(t_k))}$ from (4.5). By the central limit theorem, the Gaussian noise model (4.6) provides a close approximation to the scaled binomial model (4.4). For this reason, we adopt the simple Gaussian model of (4.6) in our numerical experiments. Section B.2 discusses how to allocate shots across time, and finds that assigning an equal number of shots to each time step performs nearly optimally.

4.2.3 Fourier data interpretation

Upon expanding (4.2) in the eigenbasis of the Hamiltonian H, the observables considered in this work are linear combinations of complex exponentials,

$$s(t_k) = \sum_{n=0}^{N-1} p_n e^{-iE_n t_k}, \quad t_k = k\Delta t, \tag{4.7}$$

where E_n denote the Hamiltonian eigenvalues, and p_n denote the squared overlaps between the Hamiltonian eigenstates and reference state, i.e. $p_n = |\langle \phi_0 | \psi_n \rangle|^2$ with

$H|\psi_n\rangle = E_n|\psi_n\rangle$. To analyze a sequence of observables $(s(t_0), s(t_1), \ldots, s(t_K))$ in the frequency domain, we can compute its discrete Fourier transform (DFT):

$$\hat{s}(f_m) = \sum_{k=0}^{K} s(t_k) e^{-i f_m t_k}, \quad f_m = \frac{2\pi m}{(K+1)\Delta t}, \tag{4.8}$$

where f_m for $0 \leq m \leq K$ are discrete frequency bins. The spacing between neighboring bins defines the associated frequency resolution $\Delta f = 2\pi((K+1)\Delta t)^{-1}$.

Under typical discretization schemes for the sequence length $K+1$ and time step Δt, the ground state energy E_0 *does not* align exactly with any of the frequency bins f_m. Accordingly, its contribution is spread across multiple frequency bins. This spectral leakage can be exacerbated by the presence of noise. Using the above, we can write the DFT of the noisy data (4.3), which we will use below in Section 4.3.2:

$$\hat{d}(f_m) = \sum_{n=0}^{N-1} p_n \sum_{k=0}^{K} e^{-i(E_n + f_m)t_k} + \sum_{k=0}^{K} \xi(t_k) e^{-i f_m t_k}. \tag{4.9}$$

4.3 Methods

4.3.1 Observable dynamic mode decomposition (ODMD)

Observable dynamic mode decomposition (ODMD) [161] is a specialized adaptation of dynamic mode decomposition (DMD) for energy estimation, formulated in the space of quantum observables. DMD is a classical data-driven technique that extracts dominant modes of a dynamical system from real-time measurements and has demonstrated success across various applications [154, 92, 62, 58, 21, 201].

DMD assumes access to snapshots of the full state of the dynamical system. Suppose we observe a dynamical system with states $\mathbf{x}(t) \in \mathbb{R}^N$ at discrete times $t = t_0, t_1, \ldots, t_K$. Given the time snapshots $\{\mathbf{x}(t_k)\}_{k=0}^{K}$, we arrange them into a pair of time-shifted data matrices,

$$X = \begin{bmatrix} \mathbf{x}(t_0) & \mathbf{x}(t_1) & \cdots & \mathbf{x}(t_{K-1}) \end{bmatrix}, \tag{4.10}$$

$$X' = \begin{bmatrix} \mathbf{x}(t_1) & \mathbf{x}(t_2) & \cdots & \mathbf{x}(t_K) \end{bmatrix}. \tag{4.11}$$

with $X, X' \in \mathbb{R}^{N \times K}$. We see that X' is X evolved forward by one time step. The forward propagator of the dynamics is then estimated via the solution of the least-squares problem

$$A^* = \arg\min_A \|X' - AX\|_F. \tag{4.12}$$

The formal solution to (4.12) is given by $A^* = X'X^+$, with X^+ being the Moore-Penrose pseudoinverse of the data matrix X. In practice, we compute the pseudoinverse via a truncated singular value decomposition (SVD) using a threshold $\delta\sigma_{\text{max}}$, where σ_{max} is the maximal singular value of X and $\delta \in (0, 1)$. This yields the regularized DMD solution $A^*_\delta = X'X^+_\delta$, where X^+_δ represents the truncated pseudoinverse of X.

For quantum systems, however, snapshots of the full state would require access to the full many-body states $|\phi(t)\rangle$ at different times; recall that such states live in an intractably high-dimensional Hilbert space. Thus, to adapt DMD to this setting, ODMD considers scalar observables of the state at different times rather than the full state snapshots. Specifically, given a Hamiltonian H with time-evolution operator e^{-iHt}, we seek observables consisting of complex-valued overlaps $s(t_k)$ from (4.2). These can be estimated with the simulated Hadamard test mentioned in Section 4.2, which measures the real and imaginary parts of the overlap and yields a noisy time series of quantum observables $d(t_k)$ as in (4.3).

Using this noisy observable time series, we now construct time-shifted matrices by specifying a delay parameter $D \ll N$. The data matrices (4.10) and (4.11) then take the form

$$X = \begin{bmatrix} d(t_0) & d(t_1) & \cdots & d(t_K) \\ d(t_1) & d(t_2) & \cdots & d(t_{K+1}) \\ \vdots & \vdots & & \vdots \\ d(t_{D-1}) & d(t_D) & \cdots & d(t_{K+D-1}) \end{bmatrix}, \tag{4.13}$$

$$X' = \begin{bmatrix} d(t_1) & d(t_2) & \cdots & d(t_{K+1}) \\ d(t_2) & d(t_3) & \cdots & d(t_{K+2}) \\ \vdots & \vdots & & \vdots \\ d(t_D) & d(t_{D+1}) & \cdots & d(t_{K+D}) \end{bmatrix}, \tag{4.14}$$

where $X, X' \in \mathbb{C}^{D\times(K+1)}$ admit a Hankel structure. To form these matrices, we need data collected at times t_k with $k = 0, \ldots, K + D$; thus when using ODMD, all expressions of K from Section 4.2 should be thought of as $K + D$. Comparing (4.13-4.14) to (4.10-4.11), we notice that each column of our new data matrices $[d(t_j), d(t_{j+1}), \ldots, d(t_{j+D-1})]^T$ plays the role of the full state, analogous to $\mathbf{x}(t_j)$ in the classic formulation. This technique is a form of time-delay embedding, which has been studied extensively in the literature [76]. Even though the delay parameter D does not satisfy the conditions required for full state reconstruction [174, 78], it is sufficient to capture relevant spectral features.

Notably, Koopman operator theory [4, 22] reveals that eigenvalues of $A^* \in \mathbb{C}^{D\times D}$, obtained by solving (4.12) using the Hankel matrices (4.13) and (4.14), must closely approximate those of H. If $\mathcal{K}_\Delta$ denotes the Koopman push operator acting on scalar

observables g via $(\mathcal{K}_\Delta g)(\psi) = g(e^{-iH\Delta t}\psi)$, then choosing the linear functional $g(\psi) = \langle\phi_0|\psi\rangle$ and initializing at $\psi = \phi_0$ yields $s(t_k) = \langle\phi_0|e^{-iHk\Delta t}|\phi_0\rangle = (\mathcal{K}_\Delta^k g)(\phi_0)$. As shown in (4.7), this expands into a sum of complex exponentials, i.e., a finite/low-rank mixture of Koopman eigenmodes with eigenvalues $e^{-iE_n\Delta t}$. The time-delay embedding with delay D builds the Krylov subspace $\mathcal{U} = \text{span}\{g, \mathcal{K}_\Delta g, \ldots\}$. The restriction $\mathcal{K}_\Delta|_\mathcal{U}$ in the Hankel coordinates of (4.13) and (4.14) is represented by a finite-dimensional shift/companion matrix whose eigenvalues are precisely the active $e^{-iE_n\Delta t}$ with non-zero overlap probabilities $p_n \neq 0$. Consequently, though we do not observe the full state, delay-embedding the scalar observables yields an invariant subspace that retains the same spectral content that we seek to estimate. As a result, ODMD then recovers the Koopman spectrum associated with $\mathcal{U}$.

Note that A^* serves as a proxy for the time evolution operator $e^{-iH\Delta t}$ whose eigenvalues are $e^{-iE_n\Delta t}$; since we approximate A^* by the truncated SVD solution $A_\delta^* = X'X_\delta^+$, the ODMD estimate of the GSE is

$$\tilde{E}_0 = -\max_{1\leq\ell\leq D} \frac{\text{Im}\log(\lambda(A_\delta^*)_\ell)}{\Delta t}, \tag{4.15}$$

where $\lambda(A^*)_\ell$ denotes the ℓ-th eigenvalue of A^*. The procedure for ODMD energy estimation is shown in the bottom left panel of Fig. 4.1, where $R = 0$.

We emphasize that, aside from obtaining the entries to populate the data matrices X, X', every computation carried out in ODMD is completely classical. Importantly, neither $A^* = X'X^+$ nor $A_\delta^* = X'X_\delta^+$ are ever loaded onto quantum hardware, so their non-unitarity has no practical impact. In the ideal, noise-free case without SVD truncation, $A^* = X'X^+$ minimizes $\|X' - AX\|_F$ and solves (4.12). When the underlying dynamics are unitary and X is well-conditioned and spans an invariant subspace, the eigenvalues of A^* tend to cluster near the unit circle, rather than lie exactly on it. In practice, measurement noise in X', X introduces projection bias and attenuates small singular vector components, which further breaks exact unitarity. Standard perturbation theory, however, shows that $\|A_\delta^* - A^*\|_F = O(\delta)$ (see (4.30)). Consequently, eigenvalues of A_δ^* remain clustered close to the unit circle; thus the imaginary part of $\log\lambda_\ell(A_\delta^*)$ still tracks the desired energy, while any small real part merely reflects damping/noise.

4.3.2 Frequency domain denoising

Fourier denoising ODMD (FDODMD) centers on two ideas, the first of which is frequency domain denoising. The denoising itself is conceptually simple: given the data (4.3), we compute its DFT (4.9) and truncate. Specifically, for a Fourier amplitude threshold $\tau_r > 0$, we truncate modes f_m for which $|\hat{d}(f_m)| < \tau_r$. We heuristically choose

$\tau_r = \gamma \cdot \text{median}(\hat{d}(f_m))$, where $\gamma > 0$ is a scalar. This choice is statistically meaningful and easily computable; setting $\gamma = 1$ provides an intuitive cutoff under reasonable assumptions on the signal-to-noise ratio. After truncating the DFT via

$$\hat{r}(f_m) = \begin{cases} \hat{d}(f_m) & |\hat{d}(f_m)| \geq \gamma \cdot \text{median}(\hat{d}(f_m)) \\ 0 & \text{otherwise,} \end{cases} \tag{4.16}$$

we apply the inverse discrete Fourier transform (IDFT) to recover the denoised signal in the time domain: for $k = 0, \ldots, K + D$,

$$r(t_k) = \frac{1}{K+D} \sum_{m \in \text{kept modes}} \hat{r}(f_m) e^{i f_m t_k}. \tag{4.17}$$

This will be referred to as the *reconstruction* or *denoised data*. The above procedure maps one time series of quantum observables $\{d(t_k)\}$ to one noise-suppressed realization $\{r(t_k)\}$. If we denoise the time series of observables multiple times—each time choosing a different threshold γ—we will obtain multiple noise-suppressed realizations.

4.3.3 Signal stacking and multi-observable ODMD (MODMD)

The second idea underpinning FDODMD is signal stacking, which uses multiple denoised realizations with ODMD to estimate the GSE. For a collection of Fourier amplitude thresholds $\{\gamma_i\}_{i=1}^R$, we first generate the ensemble of denoised realizations $\{r_i(t_k)\}_{i=1}^R$, and then stack them alongside the original noisy signal to form the following extended multi-observable vector:

$$\vec{d}(t_k) = \begin{bmatrix} d(t_k) \\ r_1(t_k) \\ r_2(t_k) \\ \vdots \\ r_R(t_k) \end{bmatrix}. \tag{4.18}$$

We modify the shifted Hankel data matrices (4.13) and (4.14) by replacing each scalar entry $d(t_k)$ with the vectorization $\vec{d}(t_k)$ defined by (4.18). This yields a pair of shifted block Hankel matrices $\mathbf{X}, \mathbf{X'} \in \mathbb{R}^{D(R+1)\times(K+1)}$ that we use to formulate $\mathbf{A}^* = \arg\min_{\mathbf{A}} \|\mathbf{X'} - \mathbf{A}\mathbf{X}\|_F$, a least squares problem. We compute the approximate solution $\mathbf{A}_\delta^* = (\mathbf{X'}\mathbf{X}_\delta^+) \in R^{D(R+1)\times D(R+1)}$ via the same truncated SVD scheme as in ODMD; our GSE estimate is (4.15) with $\mathbf{A}_\delta^*$ in place of A_δ^*. The above constitutes MODMD, a multi-observable generalization of ODMD.

Fig. 4.1 provides a schematic overview of the overall FDODMD denoising procedure. Starting with quantum observable data in the upper-left panel, we proceed to the upper-right panel by denoising with R different thresholds $\{\gamma_i\}_{i=1}^{R}$. This yields R different denoised observable realizations, which can then be stacked. For the purposes of following the FDODMD procedure itself, one can skip from the upper-right panel to the lower-left panel. The stacked denoised realizations are used to assemble the block Hankel matrices $\mathbf{X}$ and $\mathbf{X}'$, which are then used to solve for $\mathbf{A}_\delta^*$ and produce the FDODMD estimate of the GSE.

In the lower-right panel of Fig. 4.1, we illustrate both reconstruction and thresholding. On the thresholding side, for one time series of quantum observables $\{d(t_k)\}$, we plot the magnitude of the DFT $|\hat{d}(f_m)|$ as a function of frequency. When we apply thresholding as in (4.17), sub-threshold modes are truncated to zero; super-threshold modes are kept without modification. Not all discarded frequency components correspond purely to noise, implying a tradeoff between reconstruction fidelity and noise suppression that we analyze in Section 4.8. On the reconstruction side, we plot three time series: the *noiseless signal* $\{s(t_k)\}$, the *noisy data* $\{d(t_k)\}$, and the *denoised data* $\{r(t_k)\}$. Note that the denoised data is closer to the noiseless signal than it is to the noisy data, showing the utility of our simple DFT-based denoising strategy.

4.3.4 Zero-padding in time domain

As an alternative to the signal subspace method, we can enhance the performance of DFT-based estimation through a simple modification. In order to increase the frequency resolution of DFT *without* altering the original signal content, we employ zero-padding, a useful technique in signal processing [66]. Specifically, we augment our noisy time series by appending M zeros,

$$d(t_k) = \begin{cases} d(t_k), & 0 \leq k \leq K+D, \\ 0, & K+D < k \leq K+D+M. \end{cases} \tag{4.19}$$

This can be understood as modulating a time series of length $K+D+M+1$ with a rectangular window of length $K+D+1$. The DFT of this zero-padded time series becomes,

$$\hat{d}(f'_m) = \sum_{k=0}^{K+D+M} d(t_k) e^{-if'_m t_k}, \tag{4.20}$$

with refined frequency bins $f'_m = 2\pi m((K+D+M+1)\Delta t)^{-1}$ for $m = 0, \ldots, K+M+D$. We estimate the GSE by identifying the f'_m that maximizes $|\hat{d}(f'_m)|$. This refinement

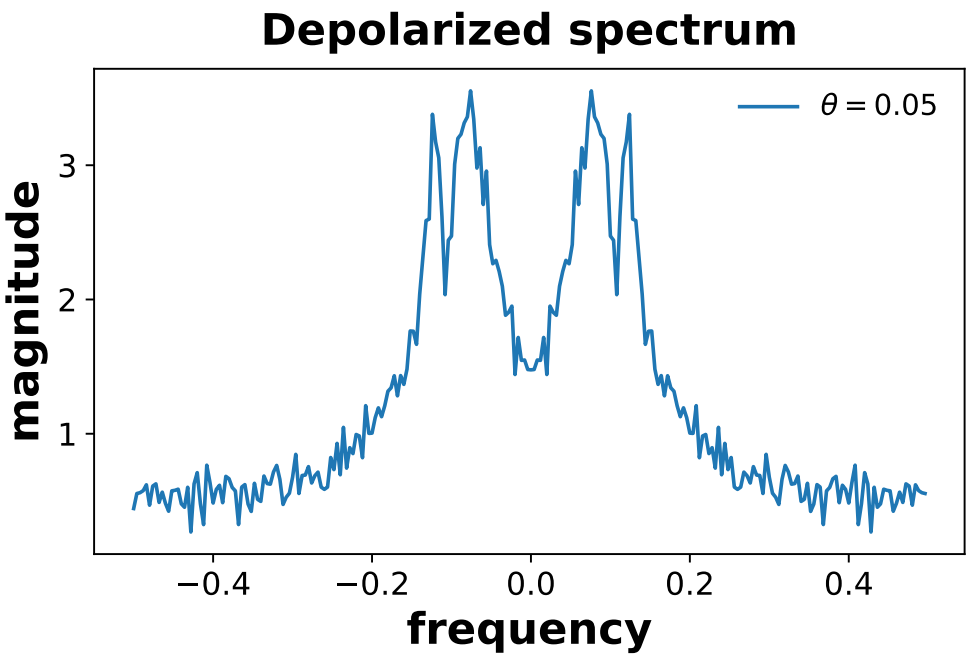
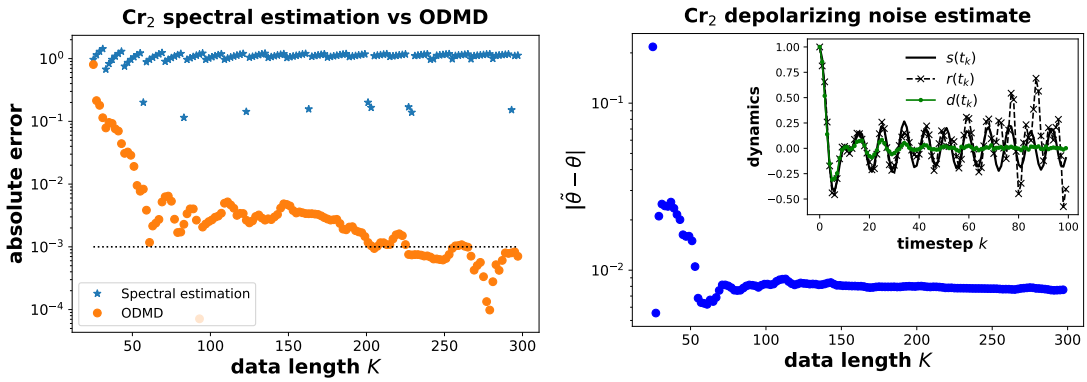


Figure 4.2: In the presence of depolarizing error and shot noise, sharp peaks in the frequency domain are obfuscated. We plot the magnitude of the DFT of the noisy time series generated from (4.22) with $(\theta, \Delta t, K_{\max}, p_0) = (0.05, 1.0, 300, 0.2)$ and $\xi(t) \sim \mathcal{N}(0, \epsilon = 0.01)$.

improves frequency resolution at no additional quantum cost, which potentially enables more accurate energy estimation.

Here we do not have the liberty of decreasing our time step $\Delta t \to 0$ to exactly resolve the GSE. The limitation arises for two reasons. First, reducing Δt would require a proportionally larger number of quantum queries to reach a given evolution time, which is infeasible under limited quantum resources. Second, ODMD and FDODMD rely on capturing dynamical features over time, necessitating a sufficiently large Δt. This is in contrast to the smaller Δt typically favored for DFT-based frequency estimation. To ensure a fair comparison, we thus keep Δt fixed across methods, including zero-padded DFT.



(a) GSE estimate

(b) Depolarizing parameter estimate and dynamics reconstruction

Figure 4.3: ODMD displays advantages in estimation and denoising for Cr_2 under depolarizing error and shot noise. See the main text in Section 4.5 for details on data generation. To compute absolute errors as a function of data length K, we follow the testing protocol described in Section 4.4. **(Left)** Convergence behavior of basic DFT-based spectral estimation (stars) and ODMD (circles). The dashed black line indicates chemical accuracy (absolute error $< 10^{-3}$.) ODMD converges quickly to chemical accuracy, in stark contrast to spectral estimation. **(Right)** Convergence of the estimate $\tilde{\theta}$ of the global depolarizing error parameter θ using ODMD. Inset in the upper right is the noisy data $\{d(t_k)\}$, the underlying signal $\{s(t_k)\}$, and the ODMD-reconstructed dynamics $\{r(t_k)\}$. See Section 4.5 for details on how the reconstruction is computed. We see favorable agreement between the reconstruction and the noiseless signal, with deviations at long times due to imperfect denoising.

4.4 Data

We use classical methods and hardware to simulate noisy quantum observables. In Sections 4.5 and 4.7, we present results for the chromium dimer (Cr_2) in the def2-SVP basis set [187]. For tractability, we truncate the Cr_2 Hilbert space via adaptive sampling configuration interaction (ASCI) [178], selecting 4000 Slater determinants with the largest coefficients. Due to strong electron correlation, Cr_2 is a stringent test for evaluating GSE methods [95, 150]. In Section B.4, we show results for lithium hydride (LiH) in the 3-21g basis set [13].

To simulate molecular data via (4.3), a key ingredient is an $N \times N$ Hamiltonian matrix H. We begin with the ASCI Hamiltonian matrix $\overline{H}$. In order to satisfy an ODMD convergence criterion [161], we apply a simple linear transformation to compute

$H = \beta_0 I + \beta_1 \overline{H}$. We detail and justify this transformation in Section B.3.

In this work, we use the time step $\Delta t = 1.0$. Let $|\psi_n\rangle$ denote the n-th eigenvector of H; note that H and $\overline{H}$ share the same eigenvectors. For $p_0 \in (0, 1)$, we prepare a reference state with ground state overlap p_0 via

$$|\phi_0\rangle = \sqrt{p_0}|\psi_0\rangle + \sum_{n=1}^{N-1} \sqrt{\frac{1-p_0}{N-1}}|\psi_n\rangle. \tag{4.21}$$

With H, Δt, and $|\phi_0\rangle$ in hand, we compute the signal (4.2) from $k = 0$ to $k = K_{\text{max}}$. Taking the real part of the signal and adding noise with *standard deviation* ϵ, we obtain (4.6). When constructing the Hankel (ODMD) and block Hankel (FDODMD) data matrices from Section 4.3, we set the delay parameter to $D = \lfloor (K+1)/2 \rfloor$. Using any of these methods, once we produce an estimate E_0^{est} of the GSE of H, we apply the inverse of the above linear transformation to produce $(E_0^{\text{est}} - \beta_0)/\beta_1$, an estimate of the GSE of the true molecular system.

Testing Protocol. Once we have produced a full data trajectory $\mathcal{D} = \{d(t_k)\}_{k=0}^{K_{\text{max}}}$, we choose K such that $K + D \leq K_{\text{max}}$ and restrict attention to $\mathcal{D}_K = \{d(t_k)\}_{k=0}^{K+D}$, the part of $\mathcal{D}$ needed to form the Hankel matrices X, X' in (4.13-4.14) or their FDODMD counterparts. In this way, we simulate the situation in which we collect only $K + D + 1$ data points to begin with. *For each algorithm, we use only the data $\mathcal{D}_K$ to estimate the GSE and quantify the error between estimated and true GSEs. Repeating this for the increasing sequence $K \in \{5, 10, 15, \ldots\}$, we quantify convergence of the GSE as a function of K, for each algorithm.* For simplicity, in the rest of this chapter, we refer to K as the *data length.*

4.5 Motivating Example

To further motivate the use of ODMD, we examine the efficacy of both ODMD and classic DFT-based spectral estimation on data corrupted with two forms of error. In addition to shot noise, we account for global gate error accumulated during time evolution. The most applicable and analytically tractable model for such error is the depolarizing channel, which captures the stochastic state relaxation to the maximally mixed state with certain probability, hence erasing quantum information [46]. This erasure noise in turn leads to an exponential damping of the sinusoidal time signal,

$$d_\theta(t) = e^{-\theta t}\langle\phi_0|e^{-iHt}|\phi_0\rangle + \xi(t), \quad \theta \in \mathbb{R}, \tag{4.22}$$

where θ characterizes the strength of the global depolarizing channel. The exponential decay reflects the loss of quantum coherence over time and sets a practical limit on the

timescale over which high-fidelity measurements can be collected from a quantum device. Furthermore, since the depolarizing noise channel effectively describes the cumulative impact of device-level errors as a uniform attenuation of the signal, it provides a useful benchmark for systematically evaluating and comparing quantum hardware performance under realistic conditions.

The Gaussian noise level ϵ associated with the statistical error term $\xi(t)$ of (4.22) captures finite-shot fluctuations of the Hadamard test. If $\hat{d}_k$ estimates $\mathrm{Re}\, s(t_k)$ from n_k shots with empirical success probability $\hat{\pi}_k \in [0,1]$, then with $\pi_k := \mathbb{E}[\hat{\pi}_k] = (1+\mathrm{Re}\, s(t_k))/2$, we have

$$\mathrm{Var}(\hat{d}_k) = \frac{4\pi_k(1-\pi_k)}{n_k} = \frac{1-(\mathrm{Re}\, s(t_k))^2}{n_k} \leq \frac{1}{n_k}.$$

Under uniform shot allocation $n_k \equiv N_{\mathrm{shot}}/K$, this justifies the shot noise level of $\epsilon \approx N_{\mathrm{shot}}^{-1/2}$ in our experiments. The global damping rate θ can associated with a discrete-time depolarizing channel. This can be realized by inserting, after each interval Δt, a per-step depolarizing channel $\mathcal{E}_{p_{\Delta t}}(\rho) = (1-p_{\Delta t})\rho + p_{\Delta t} I/D$ (where I/D is the maximally mixed state on the D-dimensional Hilbert space). Here, $p_{\Delta t} \in [0,1)$ denotes the per-step depolarizing probability. This induces a multiplicative amplitude decay $(1-p_{\Delta t})^k = e^{-\theta k \Delta t}$, yielding

$$\theta = -\frac{1}{\Delta t}\log(1-p_{\Delta t}) \approx \frac{p_{\Delta t}}{\Delta t}, \qquad (p_{\Delta t} < 1).$$

In general, the DFT constitutes a natural and convenient approach to extract energy information by decomposing a time domain signal into different frequency components, which reflect the eigenstate contributions encoded in the reference state. In noisy settings, however, spectral broadening can obscure the sharp frequency peaks, hence making it difficult to reliably estimate the GSE. This effect is particularly problematic when the evolution time is short, as spectral leakage from limited frequency resolution only exacerbates the challenge of isolating individual spectral features.

We illustrate this through an extended example. To begin, we generate data $\mathcal{D} = \{\mathrm{Re}\, d_\theta(k\Delta t)\}_{k=0}^{K_{\max}}$ using the model (4.22) with Gaussian shot noise $\xi(t) \sim \mathcal{N}(0, \epsilon = 0.01)$, depolarizing error with $\theta = 0.05$, reference state $|\phi_0\rangle$ with overlap $p_0 = 0.2$, time step $\Delta t = 1.0$, and $K_{\max} = 300$. All other details of data generation are as described in Section 4.4.

We plot in Fig. 4.2 the magnitude of the DFT of the generated time series. The DFT of a real sequence is symmetric about the zero frequency. For positive frequencies, we see that there is no single dominant peak, as we would have obtained had we switched off both shot noise and depolarizing error. The spectrum shows broad envelopes in which signal and noise intermingle, complicating GSE identification.

Continuing with the data $\mathcal{D}$, we study the convergence of the GSE estimation error incurred by two algorithms, baseline ODMD and a simple DFT-based method. By baseline ODMD, we mean ODMD from Section 4.3.1 with no frequency domain denoising or signal stacking. In the DFT-based method, we identify the discrete frequency bin f_m that maximizes the absolute value of the DFT $|\hat{d}_\theta(f_m)|$.

We apply the testing protocol described in Section 4.4 and plot in the left panel of Fig. 4.3 the absolute error of the estimated GSE as a function of data length K. ODMD achieves rapid and steady convergence to chemical accuracy (dashed line) within short real-time evolution. DFT-based spectral estimation fails to converge; it cannot be used by itself for GSE estimation from noisy data. As we will see in Section 4.7, DFT-based spectral denoising *in concert with ODMD* yields superior GSE estimates.

In addition to GSE estimation, ODMD also enables direct estimation of the depolarizing noise strength. Expanding (4.22), we find

$$d_\theta(t) = \sum_{n=0}^{N-1} p_n e^{-(\theta+iE_n)t} + \xi(t). \tag{4.23}$$

When we apply ODMD to the time series $\{d_\theta(t_k)\}$, the eigenvalues of the estimated propagator A^* will be of the form $\tilde{\theta} + i\tilde{E}_n$. As in (4.15), the imaginary part contains our estimate of E_n, while the real part provides an estimate of θ. We apply this idea in conjunction with the testing protocol from Section 4.4. This yields the right panel of Fig. 4.3: in the main plot, we plot the absolute error between estimated ($\tilde{\theta}$) and true (θ) depolarizing parameters as a function of data length K for the same experiment in the left panel. Now suppose that we take the noisy data $\{d_\theta(t_k)\}$, apply the frequency domain denoising from Section 4.3.2, and then multiply by $e^{+\tilde{\theta}t_k}$ to reverse the depolarization error and thus produce an estimate $\{r(t_k)\}$ of the original underlying signal $\{s(t_k)\}$. In the inset of the right panel of Fig. 4.3, we plot the original signal s, the noisy data d, and our estimate r. At short times, the reconstructed signal r well approximates the noiseless signal s. Deviations at longer times arise due to noise corruption, which we deal with in Section 4.7.

4.6 Practical considerations for NISQ hardware

For our numerical experiments, all real-time overlaps (4.2) are generated completely classically by forming the matrix exponential of the truncated ASCI Hamiltonian [178]. This choice isolates the algorithmic effects of the methods discussed/developed in this chapter. Crucially, this *does not* presuppose access to the underlying eigensystem on the actual quantum hardware. On a NISQ device, our workflow would only require a single

step propagator $U_{\Delta t} \approx e^{-iH\Delta t}$ that is applied repeatedly at times $t_k = k\Delta t$. Such an operator can in fact be synthesized by first or higher-order product formulas, such as Trotter-Suzuki, without the need for prior diagonalization of the Hamiltonian H.

Let $U_{\Delta t}$ admit a product formula over a Pauli decomposition of H into Γ Paulis via $H = \sum_{\nu=1}^{\Gamma} c_\nu P_\nu$. With a first-order formula and M Trotter steps per ODMD/FDODMD sampling step, define

$$S_1(\tau) := \prod_{\nu=1}^{\Gamma} e^{-iH_\nu \tau}, \quad H_\nu := c_\nu P_\nu, \quad U_{\Delta t} \approx [S_1(\Delta t/M)]^M. \tag{4.24}$$

Here, M sets the internal microstep $\Delta t/M$ used only for circuit compilation, whereas Δt remains the external sampling interval used for the classical post-processing. The product in (4.24) uses a fixed ordering, thus the compilation cost per sampling step is $O(M\Gamma)$. Furthermore, the ODMD/FDODMD algorithms themselves only require the resulting $U_{\Delta t}$ and do not require any explicit knowledge of M.

Using the operator norm $\|\cdot\|$, standard analysis shows that the first-order product formula error admits a commutator-scaled bound[33, 161]

$$\|[S_1(\Delta t/M)]^M - e^{-iH\Delta t}\| \leq \frac{\Delta t^2}{2M} \sum_{\mu<\nu} \|[H_\nu, H_\mu]\|. \tag{4.25}$$

In practice, Δt is chosen from signal-processing considerations (phase wrapping and spectral resolution), while M is set independently to meet a synthesis accuracy η, such as

$$\frac{\Delta t^2}{2M} \sum_{\mu<\nu} \|[H_\mu, H_\nu]\| \leq \eta \quad \Rightarrow \quad M \gtrsim \frac{\Delta t^2}{2\eta} \sum_{\mu<\nu} \|[H_\mu, H_\nu]\|.$$

Importantly, the Trotter error can be systematically driven to zero via increasing M [33, 123]. Thus, the resulting Trotter error is a coherent, step-stationary perturbation — meaning the same per-step bias is applied at every sampling step — of $U_{\Delta t}$. This induces a small, consistent shift in the eigenphases that ODMD/FDODMD seeks to estimate from data.

Higher-order product formulas can further improve the dependence to $O(\Delta t^{p+1}/M^p)$ with similar commutator-dependent prefactors [33, 24, 216, 123]. Direct experimental evidence from frustrated spin lattice studies shows that ODMD remains stably convergent under inexact time evolution on tangible NISQ hardware [173]. In addition, Section B.1 details a deliberately pessimistic estimation on the quantum resources required for a simple example of LiH in the STO-3G basis set, motivated by recent efforts which have demonstrated the success of simulating such a molecular system on tangible and simulated quantum hardware [207, 157, 127, 77].

4.7 Results

Here we compare and evaluate FDODMD, ODMD, and DFT-based approaches for estimating the GSE of Cr_2 under shot noise. Of main concern is the error between estimated ($\tilde{E}_0$) and true (E_0) molecular ground state energies *as a function of data length* K. *To compute all convergence results below, we followed the testing protocol described in Section 4.4.* For all calculations, we use atomic units. Our aim is to achieve chemical accuracy, defined as $|\tilde{E}_0 - E_0| < 10^{-3}$ Hartrees (units of energy). This particular tolerance reflects the precision required for molecular predictions to be chemically meaningful. In the plots below, chemical accuracy is indicated by black dashed lines.

We also investigate the sensitivity of all methods with respect to hyperparameters. These include δ (the SVD threshold chosen for FDODMD and ODMD) and p_0 (the overlap between the reference state and the true ground state). We are also interested in investigating the convergence of all methods in the high noise regime ($\epsilon \geq 0.1$).

4.7.1 Accelerated convergence relative to baseline methods

We begin with a comparison of FDODMD, baseline ODMD, and DFT-based spectral estimation. We simulate noisy data (4.6) for Cr_2 with $K_{\text{max}} = 1500$, $\Delta t = 1.0$, overlap $p_0 = 0.2$, and noise standard deviation $\epsilon = 0.1$. We then apply the testing protocol from Section 4.4 to study the convergence (as a function of data length K) of three GSE estimation methods: (i) ODMD applied directly to the noisy data, (ii) simple DFT-based spectral estimation (described in Section 4.5), and (iii) FDODMD. For FDODMD, we use the noisy data to generate $R = 6$ denoised trajectories using (4.17) with thresholds $\gamma \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5\}$. We stack the noisy data with these denoised realizations and, using the procedure from Section 4.3.3, compute the FDODMD estimate of the GSE. For FDODMD and ODMD, we use the SVD threshold $\delta = \epsilon$.

Although effective in many scenarios, baseline ODMD alone cannot completely overcome difficulties posed by noise and, moreover, hyperparameter sensitivity. Fig. 4.4 illustrates these limitations. In certain regimes (i.e. noise standard deviation ϵ, reference state overlap p_0, SVD threshold δ), basic spectral estimation fails to converge (and is thus omitted from comparisons in Figs. 4.5 to 4.7), while ODMD requires large amounts of data (here $K \geq 1000$) to converge reliably. This reveals sensitivity to hyperparameter choice and increased quantum computational cost, both of which are undesirable. By contrast, our contribution in this chapter, FDODMD, achieves chemical accuracy with faster convergence and improved robustness.

Since the algorithmic performance in the denoised framework is determined by the overall signal subspace rather than any individual time series, we can simply sweep over

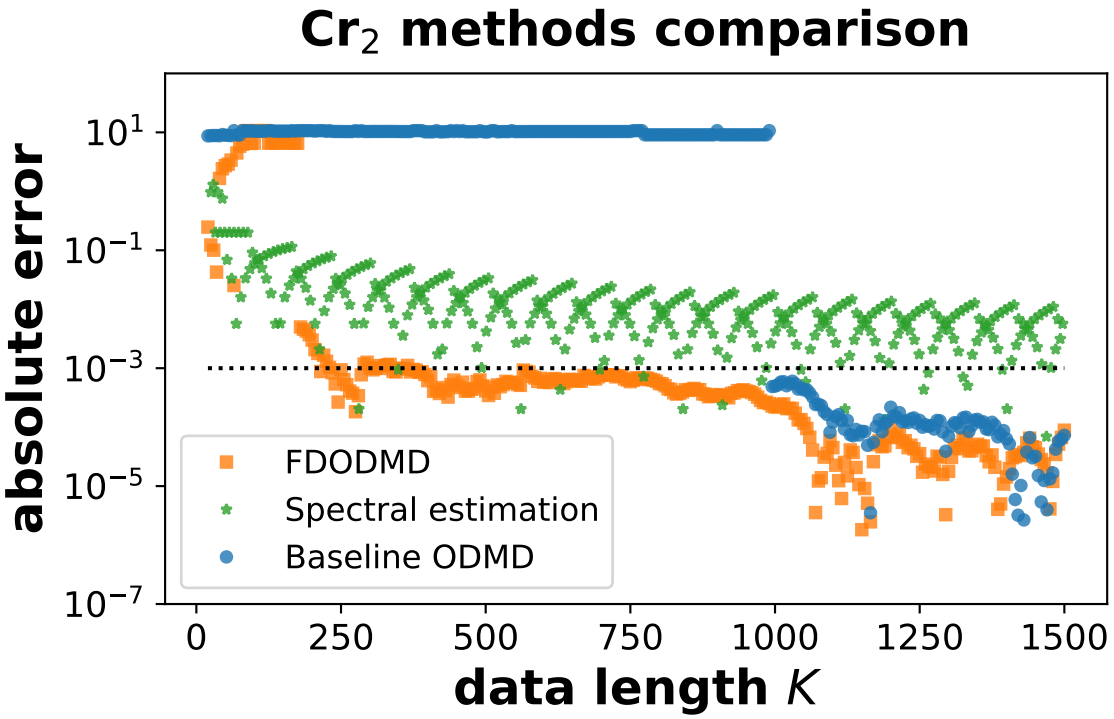


Figure 4.4: For a Cr_2 problem setting (low overlap $p_0 = 0.2$ and moderate noise standard deviation $\epsilon = 0.1$) where both baseline ODMD (circles) and DFT-based spectral estimation (stars) fail to converge (e.g., for data length $K \leq 1000$), we see that FDODMD (squares) converges. Convergence is defined as estimating the GSE to within 10^{-3} (dashed line) of its true value. Data and algorithm parameters are detailed in Section 4.7.1.

a range of thresholds $\{\gamma_i\}$ without needing to fine-tune a specific value. A distinctive advantage of our approach is that all denoised realizations $\{r_i(t_k)\}$ are generated classically from a single noisy observable trajectory, incurring no additional quantum cost, in circuits or measurements.

4.7.2 Accelerated convergence with increased hyperparameter robustness

Next we compare FDODMD and baseline ODMD for Cr_2, using reference states with varying levels of overlap p_0 with the true ground state; for both methods, we use the SVD threshold $\delta = \epsilon$. We use the same data generation parameters as in Section 4.7.1.

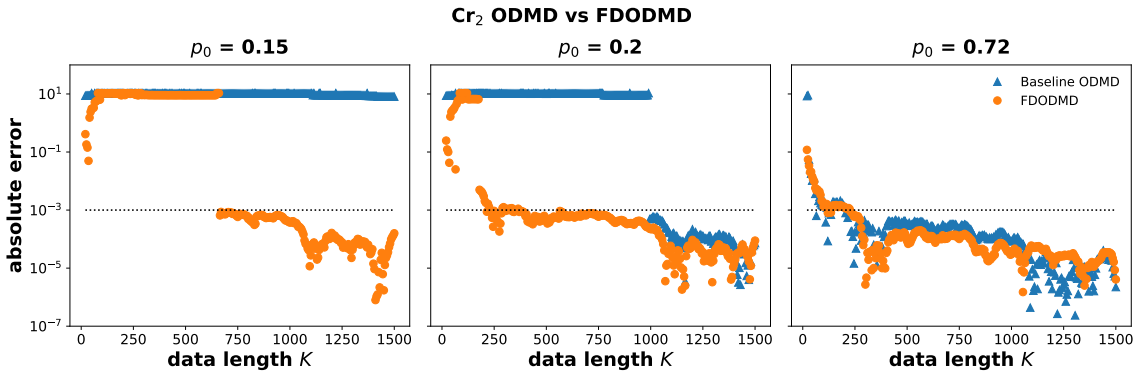


Figure 4.5: Convergence comparison of FDODMD (circles) and baseline ODMD (triangles) for Cr_2. We choose $K_{\max} = 1500$, $\Delta t = 1.0$, and $p_0 \in \{0.15, 0.2, 0.72\}$. Both methods process noisy data given by (4.6) with $\xi(t_k) \sim \mathcal{N}(0, \epsilon = 0.1)$. For FDODMD, we use (4.17) with $\gamma \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5\}$ to generate 6 denoised realizations. All the denoised realizations are then stacked in conjunction with the noisy data according to (4.18). For both FDODMD and ODMD, we set $\delta = \epsilon$. Note the accelerated convergence to chemical accuracy offered by FDODMD for the lower overlap cases.

Having already generated a trajectory with reference state overlap $p_0 = 0.2$, we generate two additional trajectories with respective overlaps $p_0 = 0.15$ and $p_0 = 0.72$; for each new trajectory, we denoise via (4.17) with thresholds given in Section 4.7.1. This yields six denoised realizations for each of three noisy data time series.

We see from Fig. 4.5 that in the low ground state overlap regime (e.g. $p_0 < 0.5$), FDODMD consistently outperforms ODMD in convergence behavior. ODMD either requires a large data length (often exceeding 1000) or even fails to reach chemical accuracy with maximal data length $K_{\max}$. In contrast, FDODMD not only converges across all tested reference states but also does so significantly faster, requiring approximately 2.5 times less quantum data than ODMD. This highlights the quantum resource savings enabled by denoising and stacking. For high ground state overlap ($p_0 = 0.72$), both approaches yield comparable convergence results. While quantum algorithms typically perform better with reference states that have a larger ground state overlap with the ground state, preparing such states can be costly [107]. The corresponding plot for LiH is shown in Section B.4, Fig. B.3.

The use of an SVD threshold δ is essential for both ODMD and FDODMD: without it (i.e. $\delta = 0$), our experiments failed to converge. To test the δ-dependence of both methods, we generate a new realization of the noisy Cr_2 trajectory with $p_0 = 0.2$ described in Section 4.7.1. For each $\delta \in \{0.15, 0.1, 0.08\}$, we test the convergence of both

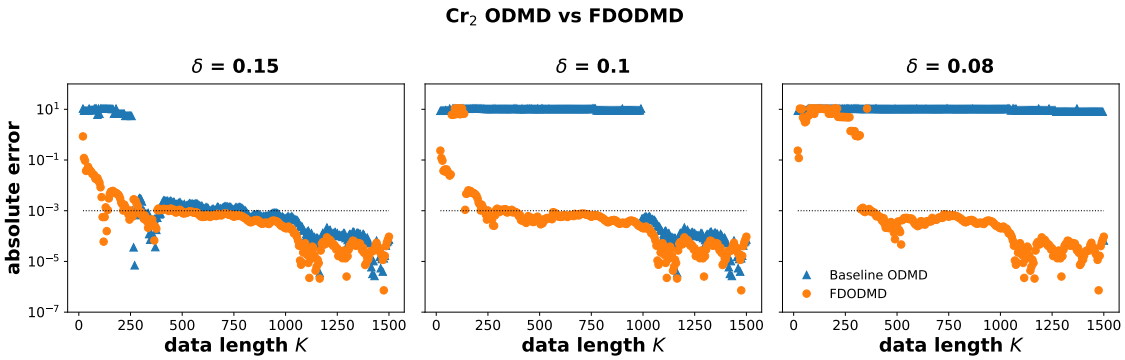


Figure 4.6: Robust invariance to SVD tolerance using FDODMD (squares) compared to SVD tolerance dependent ODMD (circles) for Cr_2. We choose $K_{\max} = 1500$, $\Delta t = 1.0$, and $p_0 = 0.2$. For both FDODMD and ODMD, we vary the SVD threshold $\delta \in \{0.15, 0.1, 0.08\}$. Both methods process noisy data given by (4.6) with $\xi(t_k) \sim \mathcal{N}(0, \epsilon = 0.1)$. For FDODMD, we use (4.17) with $\gamma \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5\}$ to generate 6 denoised realizations. All the denoised realizations are then stacked in conjunction with the noisy data according to (4.18).

ODMD and FDODMD. The results, shown in Fig. 4.6, provide evidence that moderate variations in δ do not strongly affect the performance of FDODMD. This is in contrast to baseline ODMD, which is highly sensitive to the choice of δ. The observed hyperparameter insensitivity of FDODMD is a very practical advantage, as the optimal δ is typically difficult to determine in advance without full noise characterization.

Fig. 4.7 investigates how the performance of FDODMD changes with the number of denoised trajectories. We generate a new noisy data realization with parameters $(K_{\max}, \Delta t, p_0, \epsilon) = (1500, 1.0, 0.2, 0.1)$ for Cr_2, and sequentially stack the denoised trajectories generated with $\gamma \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5\}$, assessing how FDODMD convergence improves as we include more trajectories. Black crosses display the convergence of baseline ODMD. For both ODMD and FDODMD, we choose $\delta = \epsilon$. We find that stacking more denoised signals markedly accelerates convergence to chemical accuracy, even though all methods eventually converge in the large data limit. This improvement presumably stems from the expanded signal subspace, which more effectively captures the underlying dynamics.

In total, the results in Figs. 4.5 to 4.7 demonstrate FDODMD's robustness to algorithmic hyperparameters, which allow more rapid and stable GSE estimation compared to ODMD.

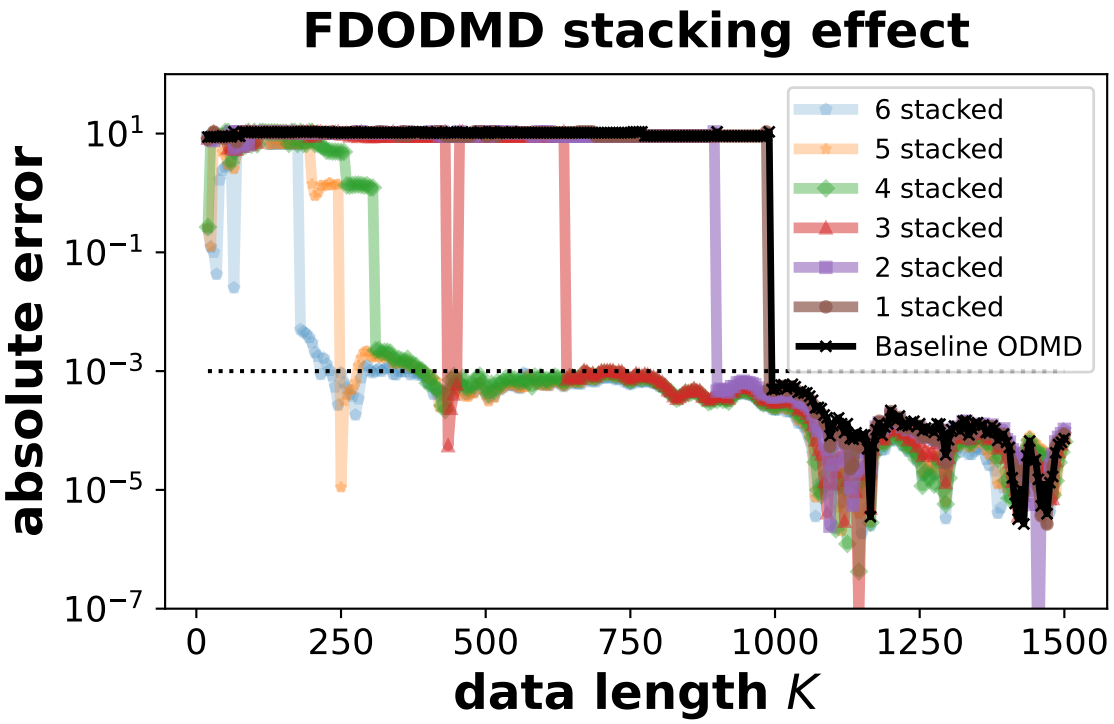


Figure 4.7: Stacking with more denoised realizations improves FDODMD convergence. We set $(K_{\max}, \Delta t, p_0) = (1500, 1.0, 0.2)$ and apply Gaussian shot noise $\mathcal{N}(0, \epsilon = 0.1)$ to simulate noisy data via (4.6). Denoised signals are then generated through (4.17) with $\gamma \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5\}$. In the plot, *1 stacked* corresponds to FDODMD where we stack the noisy data with $R = 1$ denoised realization ($\gamma = 1.0$); *2 stacked* corresponds to FDODMD where we stack the noisy data with $R = 2$ denoised realizations ($\gamma \in \{1.0, 1.5\}$), etc. Baseline ODMD is applied directly to the noisy data.

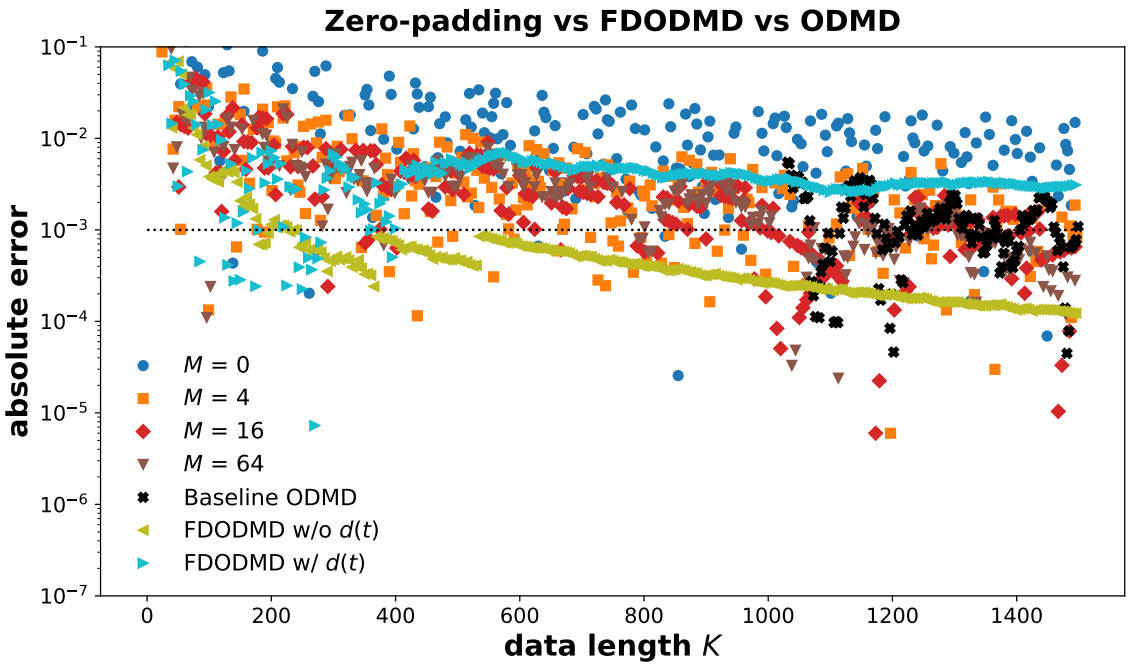


Figure 4.8: Convergence comparison of FDODMD, ODMD, and zero-padded DFT spectral estimation for the GSE of the Cr_2 molecule in the high noise regime ($\epsilon = 0.8$). For zero-padding, we append zeros equal to $M \in \{0, 4, 16, 64\}$ times the length of the original time series, and perform DFT-based frequency estimation as in (4.20). For both FDODMD and ODMD, we choose $\delta = \epsilon$ as the SVD parameter. We generate noisy data using (4.6) with $\xi(t_k) \sim \mathcal{N}(0, \epsilon = 0.8)$ and $(K_{\max}, \Delta t, p_0) = (1500, 1.0, 0.2)$. For FDODMD, we use (4.17) with $\gamma \in \{2.0, 2.5, 3.0, 3.5, 4.0, 4.5\}$ to generate $R = 6$ denoised realizations. FDODMD (excluding noisy data from the stack) achieves the most stable and accurate convergence; in this high noise regime, stacking *with* the noisy data is detrimental for GSE estimation.

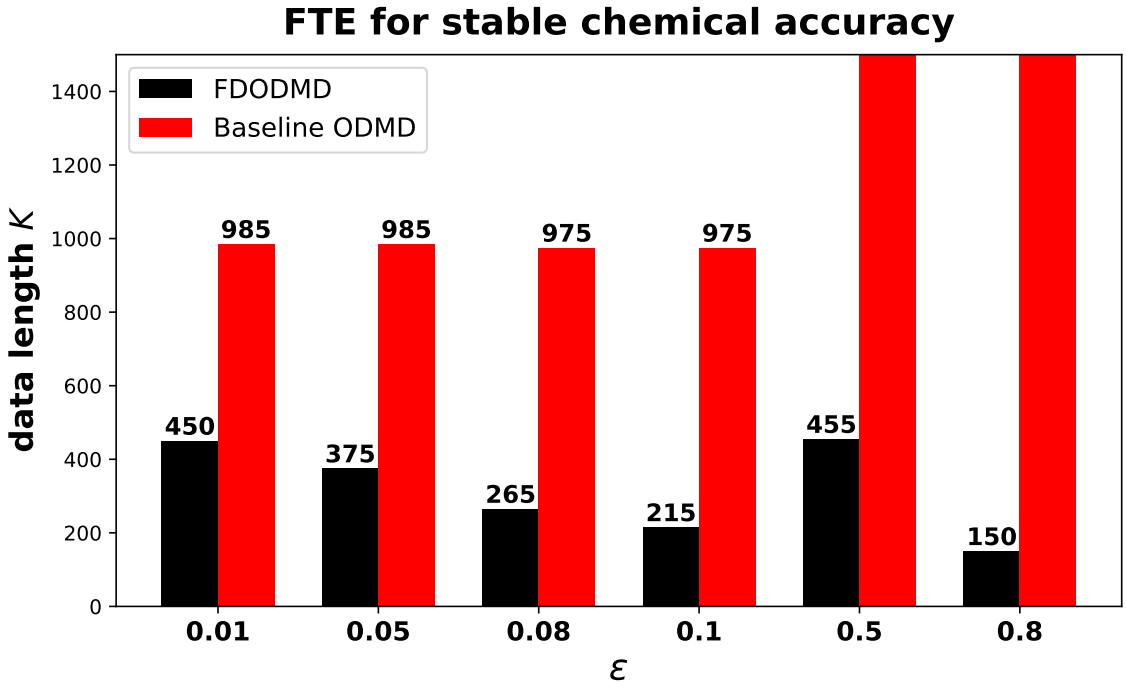


Figure 4.9: Stability of FDODMD under increasingly higher noise. For each value of ϵ, we generate noisy data using (4.6) with $\xi(t_k) \sim \mathcal{N}(0, \epsilon)$ and $(K_{\max}, \Delta t, p_0) = (1500, 1.0, 0.2)$. We report the data length required to reach stable chemical accuracy for Cr_2 molecule. ODMD processes the noisy data directly, whereas FDODMD uses a stack of $R = 8$ denoised realizations constructed with $\gamma \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5\}$. We choose the SVD threshold $\delta = \epsilon$. The noisy data is excluded from the stack. For all noise standard deviations ϵ, FDODMD uses fewer quantum resources (shorter bar). For the highest noise cases ($\epsilon = 0.5, 0.8$), baseline ODMD does not converge stably to chemical accuracy with data length within $K_{\max}$.

4.7.3 Convergence in the high noise regime

In Fig. 4.8, we compare FDODMD, baseline ODMD and the zero-padding procedure described in Section 4.3. For Cr_2, we generate noisy data with parameters $(K_{\max}, \Delta t, p_0) = (1500, 1.0, 0.2)$ and noise standard deviation $\epsilon = 0.8$; we vary the zero-padding factor $M \in \{0, 4, 16, 64\}$ in (4.20), where $M = 0$ corresponds to standard DFT frequency estimation. While higher padding factors ($M = 16, 64$) allow zero-padding to outperform ODMD in convergence speed, FDODMD still achieves superior stability and accuracy, as noise is taken into consideration. In addition, we assess the impact of incorporating the original noisy data into the FDODMD stacking procedure. As illustrated in Fig. B.4 in Section B.4, the convergence of the estimated GSE is evaluated for both cases. When the noisy data is included, the results exhibit clear deviations from standard bounds [136, 118], thereby indicating the need to exclude it for stable convergence. This procedure is practically feasible as the classical cost of running FDODMD multiple times is cheap. It is important to note that the question of whether to include the noisy data in the stack is unique to the high noise regime ($\epsilon \geq 0.5$).

As the noise standard deviation ϵ increases, the stability of baseline ODMD deteriorates, particularly for short time evolution. In Fig. 4.9, we compare the performance of ODMD and FDODMD on Cr_2 molecule with varying ϵ. For FDODMD, we use $R = 8$ denoised trajectories with $\gamma \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5\}$, this time excluding the original noisy data. This exclusion is motivated by the results shown in Fig. 4.8. For both methods, we fix $(p_0, \Delta t, K_{\max}) = (0.2, 1.0, 1500)$ and report in Fig. 4.9 the data length required to achieve chemical accuracy for at least 10 consecutive values of K (data length). Here taller bars indicate longer data lengths (i.e. higher quantum costs). When noise is high ($\epsilon = 0.5, 0.8$), baseline ODMD fails to stably reach chemical accuracy, as manifested by the bars that exceed the budget. FDODMD, in contrast, requires ≤ 455 time steps to converge. When noise is small enough ($\epsilon < 0.1$) for ODMD to converge, FDODMD also does so using notably less data — substantiating the efficiency gains from our denoising and stacking procedures.

Fig. 4.9 shows a non-monotone dependence of the required data length K on the noise level ε. This stems from our SVD truncation rule $\delta = \varepsilon$: larger ε induces more aggressive truncation that removes singular directions in the noise bulk and improves the conditioning of the least-squares fit. As a result, there can be a decrease in K at higher ε. This regularization effect is also consistent with prior DMD analyses [38, 126]. A further distinction is that FDODMD applies Fourier-domain denoising before assembling the block-Hankel matrices, creating a cleaner separation between signal- and noise-dominated directions and reducing sensitivity to the choice of δ. In comparison, ODMD lacks this prefiltering, so truncation primarily stabilizes the fit without delivering

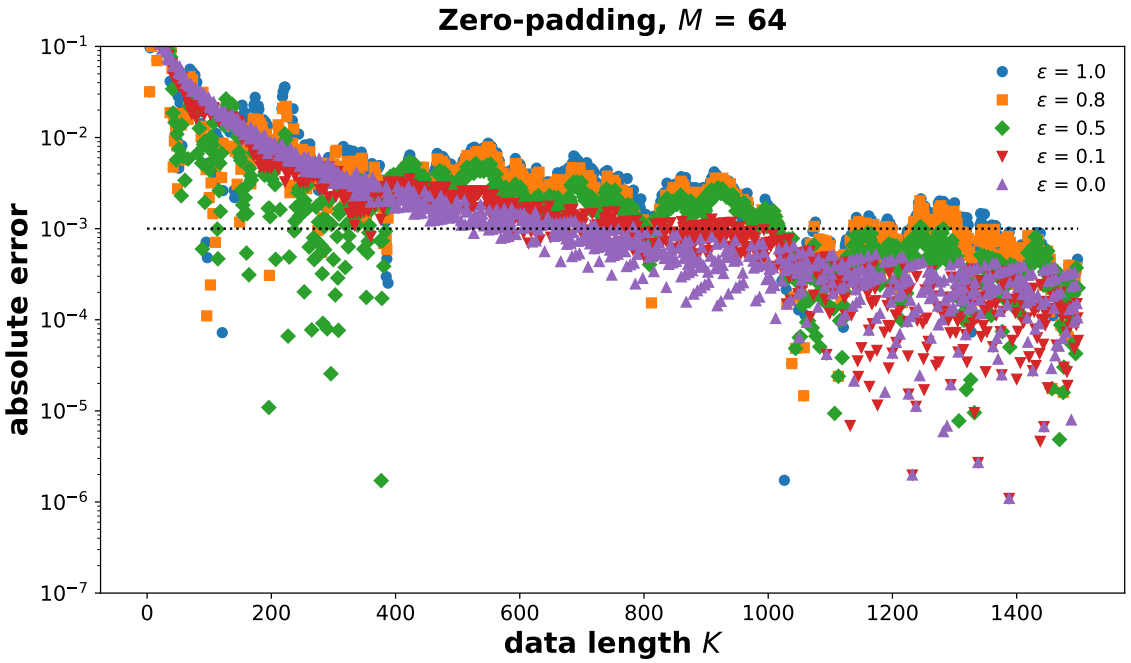


Figure 4.10: Noise robustness of zero-padding spectral estimation for the Cr_2 molecule. A fixed padding size $M = 64$ is considered and the noise standard deviation ϵ are varied. For zero-padding, we append zeros equal to M times the length of the original time series, and apply DFT-based frequency estimation as in (4.20). We generate noisy data using (4.6) with $\xi(t_k) \sim \mathcal{N}(0, \epsilon)$ and $(K_{\max}, \Delta t, p_0) = (1500, 1.0, 0.2)$. Overall, the convergence for a fixed padding factor is not strongly affected by variations in the noise standard deviation.

comparable denoising, leading to no obvious gain or decay in K-convergence.

To further examine noise robustness, Fig. 4.10 studies the performance of the zero-padding procedure with a fixed padding factor $M = 64$. We apply this procedure to noisy trajectories with varying ϵ. In comparison to FDODMD and ODMD, the zero-padding method's overall error decay (as a function of data length) remains relatively insensitive to the noise standard deviation ϵ.

In total, the results in Figs. 4.8 to 4.10 demonstrate that our methods deliver reliable convergence in the high noise regime, where baseline ODMD fails to converge within the available quantum resource budget.

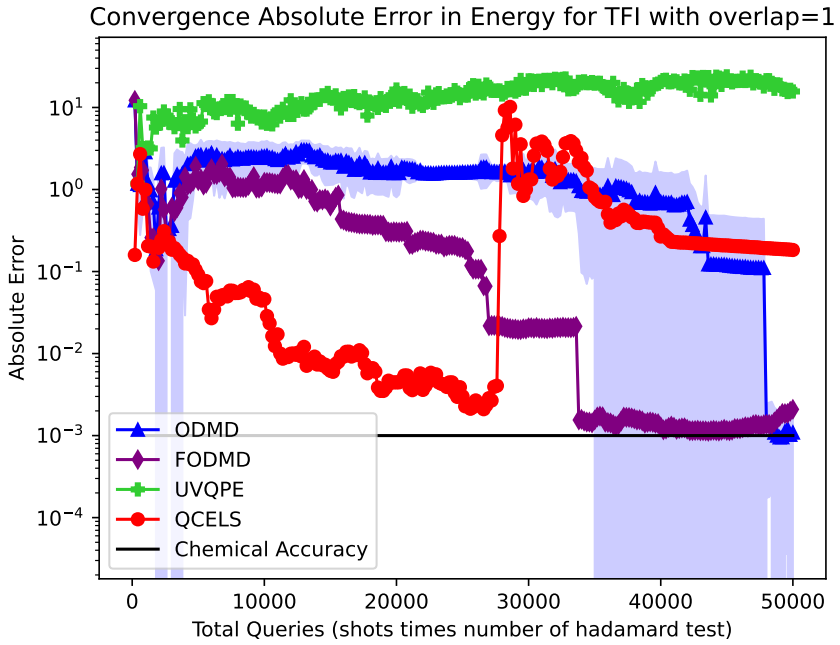


Figure 4.11: Absolute ground-state energy error versus total queries for ODMD, FDODMD, and additional hybrid baselines [88, 43] on a three-qubit transverse-field Ising chain run on `ibm_fez`. We use $Q = 3$, $g = 4$, $N = 500$ time points with step size $\Delta t = 1$, $N_s = 10^2$ shots per point, and an even-superposition initial state; FDODMD is run with thresholds $\gamma = 1, 2, 3, 4$. FDODMD converges much more stable than other GSE algorithms. (Figure and experiment courtesy of mentored students Alexander Weiss and Paul Bruzzi of Rensselaer Polytechnic Institute).

4.7.4 Execution on quantum hardware

To complement the simulation studies, we also carried out a proof-of-principle hardware experiment on the `ibm_fez` backend, using a three–qubit transverse-field Ising (TFI) chain as a minimal test system. The target Hamiltonian is

$$H_{\mathrm{TFI}} = -\sum_{j=1}^{Q-1} Z_j Z_{j+1} - g\sum_{j=1}^{Q} X_j, \tag{4.26}$$

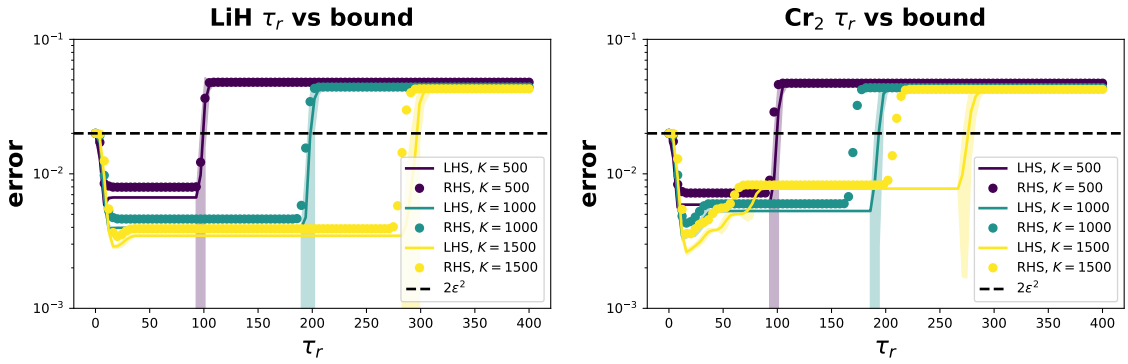
with $Q = 3$ sites (qubits) and transverse field strength $g = 4$. Here X_j and Z_j denote Pauli operators acting on the jth qubit:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

and X_j (resp. Z_j) means X (resp. Z) tensored with identity on all other qubits. The interaction term $Z_j Z_{j+1}$ couples neighboring spins along the z–axis, while the transverse-field term X_j tends to align spins along the x–axis. For each algorithm (ODMD, FDODMD, and other hybrid GSE algorithms such as UVQPE [88] and QCELS [43]), we ran Hadamard-test circuits with $N = 500$ equally spaced time points at step size $\Delta t = 1$, using $N_s = 10^2$ shots per time point; the FDODMD runs used frequency thresholds $\gamma = 1, 2, 3, 4$. Figure 4.11 plots the absolute error in the estimated ground-state energy versus the total number of queries (shots times the number of Hadamard tests). Across the accessible query budget, FDODMD (labeled FODMD) converges and attains chemical accuracy significantly more stable compared to other methods. ODMD and QCELS exhibit larger fluctuations and typically saturate above the chemical-accuracy threshold, and UVQPE does not converge at all in this noisy, low-shot regime. This hardware experiment provides an independent validation that FDODMD can trade classical post-processing for reduced quantum overhead in practice on tangible NISQ hardware, mitigating the combined bottleneck of exponential Hilbert-space dimension and noisy measurements on near-term devices.

4.8 Formal justification of FDODMD

To support and complement the numerical results of Section 4.7, we present an initial error analysis of FDODMD.



(a) LiH denoising bound.

(b) Cr_2 denoising bound.

Figure 4.12: Tightness of the bound developed in (4.27) for both LiH and Cr_2 averaged across 100 noisy trajectories. For the left-hand term (LHS), the mean is plotted in the solid line and the shaded band represents the standard deviation. This is not needed for the right-hand term (RHS) as it is deterministic. Here, for both molecular systems we fix $\epsilon = 0.1$. Note that the y-axis here is rescaled by $1/K$. For three values of K, we see that the upper bound is reasonably close to the actual denoising error. As $\tau_r \to 0$, no thresholding/denoising is applied; thus both sides of the bound in (4.27) approach $2\epsilon^2$. As $\tau_r \to \infty$, the reconstructed signal vanishes ($\mathbf{r} \to 0$); thus both sides of the bound approach $\|\mathbf{s}\|_2^2$, the squared 2-norm of the noiseless signal.

4.8.1 Analysis of denoising

In this subsection we offer error analysis of the denoising procedure detailed in Section 4.3. We defer all proof details to Section B.5. Let the noiseless, noisy, and denoised data be constructed by (4.2), (4.3), and (4.17), respectively; let $\mathbf{s}$, $\mathbf{d}$, and $\mathbf{r}$ denote their vectorized time domain representations.

Theorem 4.8.1. *The mean-squared error between the reconstructed data* $\mathbf{r}$ *and noiseless data* $\mathbf{s}$ *is bounded above by*

$$\frac{1}{K}\mathbb{E}\left[\|\mathbf{r}-\mathbf{s}\|_2^2\right] \leq 2\epsilon^2 G(\tau_r, K, \epsilon, \mathbf{s}), \tag{4.27}$$

for a function G that depends on the signal and noise characteristics. Here, we provide a proof outline; the full proof and an explicit functional form of G can be found in Section B.5. Importantly, we validate that there exist threshold choices τ_r for which $G(\tau_r, K, \epsilon, \mathbf{s}) < G(0, K, \epsilon, \mathbf{s}) \equiv 1$. Moreover, under a fixed thresholding policy, the bound tightens with K in the sense that $G(\tau_r, K, \epsilon, \mathbf{s})$ approaches a τ_r-dependent constant as K

increases, with a $1/K^2$ decay contributed by the signal-leakage term, while the threshold-controlled noise-retention term remains K-stable. Thus, the bound in (4.27) tightens with K while its limiting value is set by τ_r.

Intuitively, as K grows, the DFT's frequency resolution improves—true spectral information concentrates while noise remains diffuse—so thresholding discards noise without sacrificing signal more effectively; measured per sample, this yields a $1/K^2$ decay in signal leakage while the threshold-controlled noise contribution remains K-stable.

Proof outline. Direct calculation shows that the error between the noisy and noiseless data is

$$\mathbb{E}\left[\|\mathbf{d}-\mathbf{s}\|_2^2\right] = \mathbb{E}\left[\sum_{n=0}^{K-1}|\xi_n|^2\right] = 2K\mathbb{E}[(\mathrm{Re}\,\xi_0)^2] = 2K\epsilon^2, \tag{4.28}$$

where ξ_n are noise random variables whose real and imaginary parts are i.i.d. (independently and identically distributed) Gaussians, i.e., $\mathrm{Re}\,\xi_n \sim \mathcal{N}(0,\epsilon)$ and $\mathrm{Im}\,\xi_n \sim \mathcal{N}(0,\epsilon)$. Examining the error between the denoised and noiseless data, we see that by Plancherel's theorem,

$$\|\mathbf{r}-\mathbf{s}\|_2^2 = \frac{1}{K}\|\widehat{\mathbf{r}}-\widehat{\mathbf{s}}\|_2^2 = \frac{1}{K}\sum_{k=0}^{K-1}\mathbb{I}_{|\widehat{d}_k|\geq\tau_r}|\widehat{\xi}_k|^2 + \mathbb{I}_{|\widehat{d}_k|<\tau_r}|\widehat{s}_k|^2, \tag{4.29}$$

where $\mathbb{I}$ denotes an indicator variable. One can show that $|\hat{\xi}_k|^2$ are exponentially distributed with rate parameter $\lambda = (2\epsilon^2 K)^{-1}$. Evaluating the two terms on the RHS of (4.29), we obtain the bound $\mathbb{E}\left[\|\mathbf{r}-\mathbf{s}\|_2^2\right] \leq 2K\epsilon^2 G$ for a suitable function $G(\tau_r, K, \epsilon, \hat{s}_k)$, therefore completing the proof.

As $\tau_r \to 0$, we have $G \to 1$ so that both sides approach (4.28). As $\tau_r \to \infty$, both sides approach $\|\mathbf{s}\|_2^2$. Essentially, when no truncation is done, both sides of (4.27) approach the 2-norm of the noise, and when a total truncation of all the modes is done, both sides approach the 2-norm of signal.

Shown in Fig. 4.12 is a numerical comparison of the left and right-hand sides of (4.27) for both the molecular systems studied in this text. We fix the noise standard deviation $\epsilon = 0.1$, time step $\Delta t = 1.0$, and initial state preparation with $p_0 = 0.2$, while varying the lengths K of the observable series and truncation threshold τ_r. Numerically, we observe that the upper bound in (4.27) gives a reasonable estimate. In practice, since we set $\tau_r = \gamma_r * \text{median}(\hat{d}_n)$ with $\gamma_r \in [1, 4.5]$, the τ_r values for the reported results lie within the range of $[2, 26]$. We find in Fig. 4.12 that this range is optimal. The large vertical jumps seen in the figure arise from cases where all frequency modes in the reconstruction are truncated.

4.8.2 Analysis of the estimated GSE

If $\overline{A} = X'X^+$ denotes the solution to (4.12) using the noiseless versions of X, X', then we can define $\tilde{A} = (X' + \delta X')(X + \delta X)^+$ to be the analogous solution using the denoised data, where $\delta X, \delta X'$ represent perturbations of X, X', respectively. With this, it follows that

$$\|\widetilde{A} - \overline{A}\|_2 \leq \frac{\kappa}{1 - \kappa\eta} \left(\|\overline{A}\|_2 \eta + \|\rho\|_2 \frac{\kappa\eta}{\|X\|_2} + \frac{\|\delta X'\|_2}{\|X\|_2} \right) + \eta \|X\|_2 \|Y\|_2. \quad (4.30)$$

Here $\kappa = \|X\|_2 \|X^+\|_2$, $\eta = \|\delta X\|_2 / \|X\|_2$, $\rho = X' - AX$, and $Y = X'(X^\dagger X)^+$. The proof is a small perturbation of the arguments of Wedin [186], who considered a least squares problem of the form $a^* = \arg\min_a \|x' - Xa\|_2$, where a, x' are vectors and X is a matrix. Changing a, x' to matrices A, X', reversing the order of the product XA to AX, and replacing Wedin's vector 2-norm with the matrix Frobenius norm, one can repurpose Wedin's arguments [186] to prove (4.30). From (4.30), we see that the right-hand side tends to 0 as the perturbations $\delta X, \delta X' \to 0$.

If $\tilde{A}_1$ represents the perturbed DMD solution using the denoised data, with corresponding perturbations $\delta X_1, \delta X_1'$ and $\tilde{A}_2$ represents the perturbed DMD solution using the noisy data, with corresponding perturbations $\delta X_2, \delta X_2'$, then knowing $\|\delta X_1\|_2 \leq \|\delta X_2\|_2$ and $\|\delta X_1'\|_2 \leq \|\delta X_2'\|_2$ is not sufficient to show that $\|\tilde{A}_1 - \overline{A}\|_2 \leq \|\tilde{A}_2 - \overline{A}\|_2$. The bound in (4.30) demonstrates that if $\|\tilde{A}_1 - \overline{A}\|_2 \leq C_1$ and $\|\tilde{A}_2 - \overline{A}\|_2 \leq C_2$, then $C_1 \leq C_2$. Namely, this implies that denoising reduces an upper bound of the error between system matrices.

In practice, we know that the perturbations of the denoised data matrices and the noisy data matrices will obey $\|\delta X_1\|_2 \leq \|\delta X_2\|_2$ and $\|\delta X_1'\|_2 \leq \|\delta X_2'\|_2$. The bounds in (4.27) and (4.28) represent uniform upper bounds on the columns of $(\delta X_2, \delta X_2')$ and $(\delta X_1, \delta X_1')$, respectively. Thus, for correctly chosen truncation factors τ_r, denoising reduces an upper bound on the error of the DMD system matrices. Since the eigenvalues of the DMD system matrices exactly determine the estimated GSE, we see that the denoising procedure reduces an upper bound on the error of our estimated GSE.

4.9 Conclusion

In this chapter, we addressed the challenge of reliably estimating the ground state energy (GSE) of molecular systems from noisy quantum observables. Acknowledging that common noise sources such as depolarizing error and shot noise can significantly

degrade the performance of many spectral estimation approaches, our goal was to develop a hybrid algorithm that remains robust to a high amount of noise while exhibiting a low sensitivity to its algorithmic hyperparameters.

We introduced a novel framework — Fourier denoising ODMD (FDODMD) — which builds on observable dynamic mode decomposition (ODMD) by combining classical Fourier-based thresholding with quantum signal subspace techniques. By leveraging frequency-domain thresholding to suppress spurious spectral modes introduced by noise, FDODMD reconstructs cleaner time-domain signals from a single noisy observable trajectory. It then generates multiple denoised realizations across varying thresholds and stacks them into a higher-dimensional signal subspace, enabling the extraction of richer spectral information. This approach incurs no additional quantum cost and significantly improves convergence to chemical accuracy in high error scenarios, even in regimes where baseline ODMD fails.

Numerical experiments on challenging molecular systems, such as LiH and Cr_2, demonstrate that FDODMD consistently outperforms baseline ODMD and classical Fourier spectral estimation. The results reflect rapid energy convergence with markedly shorter simulation times, substantially lowering the quantum resources required. Moreover, our analytical bounds validate that the denoising procedure effectively reduces the reconstruction error relative to the noisy dynamics. Finally, preliminary work on tangible hardware shows promising results for execution of the developed algorithm in practice.

In summary, FDODMD provides a robust and resource-efficient algorithm that overcomes some key obstacles of GSE estimations in noisy settings. By combining classical Fourier denoising with a quantum signal subspace eigensolver, it opens up a promising avenue for the design of accurate and scalable hybrid algorithms for computational chemistry and beyond.

Chapter 5

Conclusion and future work

In this disseration, we developed a set of data-driven, equation-aware methods that target specific structural bottlenecks in time-dependent systems, and applied such methods to challenges in quantum and electron dynamics. In Chapter 2, we introduced a neural network based surrogate that replaces the expensive history integral in an IDE by a learned time-local map, reducing temporal complexity from quadratic to effectively linear while maintaining accuracy over long extrapolation horizons. Chapter 3 developed SRaFTE, a two-phase coarse to fine grid forecasting framework that couples a low-cost coarse-grid time integrator with a learned spatial SR operator. We demonstrated remarkable success on canonical PDEs and applied the methodology to plasma dynamics governed by the Vlasov–Poisson system. This resulted in accurate fine-scale reconstructions and long-time forecasts at substantially reduced runtime and storage cost. Finally, Chapter 4 proposed FDODMD, a hybrid quantum-classical post-processing pipeline that processes noisy time-evolution signals from near-term quantum hardware to estimate the ground-state energy of molecular Hamiltonians with improved robustness and reduced quantum overhead relative to existing hybrid ground-state energy estimation algorithms and spectral estimators.

Time-evolving systems and spatiotemporal dynamics lie at the heart of all three contributions of this dissertation. Across nonequilibrium quantum dynamics, kinetic plasmas, and quantum chemistry calculations executed on near-term, noisy quantum hardware, accurate simulation and inference require resolving high-dimensional state spaces, handling temporal nonlocality and memory effects, and coping with noisy or incomplete observations. Direct numerical methods that fully resolve all of the previously mentioned issues are often intractable and introducing simplifications or approximations can erase the very phenomena of interest that was set out to be studied. The overarching theme of this dissertation has been to identify the computational bottleneck in each case study and replace it with some data-driven surrogate that can be tightly coupled to the

downstream objective: whether it be propagating dynamics or estimating spectral properties.

The first case study (Chapter 2) focused on IDEs in which a nonlinear memory integral over the past trajectory dominates the the computational cost of conventional numerical solvers. There, the work showed that by training an LSTM-based RNN on modestly sized dataset of trajectories from a reference IDE solver on a short time window, one can learn an effective short-memory map that approximates the history integral term. When the learned map is substituted back into the original IDE in place of the memory integral, the resulting time-local evolution can be advanced with linear time complexity, in contrast to the typical quadratic time scaling, whilst also closely tracking the reference dynamics on the benchmark problems considered. Ultimately, this demonstrates that learning the memory functional for the class of IDEs studied can be more cost effective that direct numerical solvers and also more accurate than learning a direct propagator of the dynamics, especially when probing out-of-distribution dynamics.

The second case study (Chapter 3) addressed propagation of spatiotemporal PDEs where the bottleneck is fine-grid propagation over long times. SRaFTE was introduced as a two-phase learning framework: Phase 1 learns a coarse to fine SR map from paired coarse–fine snapshots on a short time interval, while Phase 2 embeds this learned lift inside a predictor–corrector type loop to form an effective fine-grid surrogate that never explicitly steps in time on the fine grid. On the 2D heat, wave, and Navier-–Stokes equations, SRaFTE achieved accurate SR in Phase 1 and superior long-horizon forecasts in Phase 2 compared with autoregressive neural operator baselines and naive upsampling, particularly for the nonlinear incompressible Navier-–Stokes system where error accumulation is a major challenge. The framework also yielded substantial savings in storage and wall-clock time by allowing practitioners to archive and evolve only coarse trajectories.

Chapter 3 also developed a deterministic error analysis for SRaFTE that decomposes the one-step error of the effective fine propagator into contributions from the neural lift, the coarse integrator, and projection. This led to global error bounds to quantify the error accumulation in Phase 2 expressed in terms of local Lipschitz constants and per-step approximation errors, as well as a Rademacher-style generalization bound for the learned lift trained on finite data and its generalization to out of distribution initial conditions. This error analysis and the bounds developed provide a first step toward understanding how local approximation quality from the learned SR operator and stability properties of the coarse solver interact to control long-time behavior in coupled solver–surrogate systems.

Also within Chapter 3, we applied SRaFTE to time-dependent electron dynamics governed by the Vlasov–Poisson system. Here, the challenge is to resolve fine-scale

interactions in a kinetic plasma without paying the full cost of fine-grid simulations. We use the SRaFTE framework and evaluate performance not only in pointwise error but also in plasma diagnostics, such as errors in computable observables, and show that the learned SR operator can reconstruct fine-scale phase-space structures and their moments from coarse-grid simulations with high fidelity. Phase 2 rollouts for randomized electron and two-stream instability experiments demonstrated that the SRaFTE pipeline preserves key kinetic features well beyond the training window, at appreciable reductions in runtime relative to a fine-grid solver. This case study reinforces that the same equation-aware design can translate from canonical PDEs to more realistic electron-scale systems.

The final case study (Chapter 4) shifted focus from classical forward simulation of time-evolving systems to spectral inference for quantum chemistry Hamiltonians, whose Hilbert space dimension grows exponentially with system size. In this setting, the bottleneck is twofold: dimensionality still constrains the classical resources that can be devoted to time evolving the Hamiltonian according to the TDSE, and noise limits the robustness and sample efficiency of ground-state energy estimation from the resulting time-evolved signals on quantum hardware. FDODMD was proposed as a quantum–classical pipeline that leverages ODMD [161] combined with frequency-domain denoising and signal stacking to post-process noisy time-evolution signals and extract the ground-state energy. By thresholding in the frequency domain, reconstructing multiple denoised signals across thresholds, and stacking them into an enlarged data matrix before applying ODMD, FDODMD improves the conditioning of the spectral estimation problem without additional quantum resources. Perturbation bounds showed that the denoising step reduces an upper bound on the error of the DMD system matrices and associated eigenvalues, while numerical experiments on representative molecular Hamiltonians demonstrated faster and more robust convergence to chemical accuracy compared with standard ODMD and Fourier-based estimators, especially in low-overlap and high-noise regimes, with minimal tuning of algorithmic hyperparameters such as SVD threshold.

Across these three lines of work, several common design principles emerge. First, the methods are operator-centric. They focus on approximating specific evolution, memory, or spectral operators that are dictated by the governing equations and are known to dominate computational cost. Second, they are equation-aware rather than equation-free: the IDEs, PDEs, and Schrödinger dynamics are used to decide what to learn and where to insert it into an existing numerical or hardware pipeline, but the equations are not enforced as hard constraints during training. This stance allows the learned components to remain modular and compatible with legacy solvers and currently used quantum hardware workflows. Finally, the evaluation of each developed method emphasizes both

accuracy and tractability: each method is assessed not only by approximation error but also by runtime, storage, and quantum resource requirements, reflecting the practical constraints under which real simulations and experiments operate.

The dissertation also has clear limitations that point toward future work. The surrogates are trained and tested on model problems of moderate size and regularity, and their behavior under strong distributional shift remains only partially characterized. The memory surrogates of Chapter 2 have not yet been deployed in full-scale NEGF/KBE solvers (though progress was made in [215]), and the plasma experiments in Chapter 3 assume simplified geometries and ion dynamics. Furthermore, the noise models in Chapter 4, while representative, do not capture all device-specific error channels encountered on current quantum hardware, such as gate errors. Extending these methods to more complex systems and validating them in production-grade codes and experimental settings on tangible hardware is an important next step.

Looking forward, several directions appear especially promising. On the theoretical side, extending the current error analyses to capture data-dependent generalization, interactions among multiple learned operators, and stability properties under perturbations of the equations themselves would deepen our understanding of when and why equation-aware surrogates succeed. On the methodological side, one can seek more expressive and structure-preserving architectures for memory and super-resolution operators, integrate adaptive or unstructured meshes into SRaFTE-type frameworks, and develop additional denoising/spectral-based enhancements to signal subspace-like methods. On the application side, integrating these methods into large-scale NEGF, plasma, and quantum chemistry workflows/production codes, and deploying FDODMD on NISQ quantum hardware alongside state-of-the-art error-mitigation schemes would provide a more definitive assessment of the impact of the developed methods of this dissertation.

In summary, this dissertation has shown that data-driven, equation-aware methods can substantially alleviate canonical computational bottlenecks in time-dependent systems, and show great success when applied to quantum and electron dynamics, while preserving the core physical structure encoded by the underlying equations of motion. By surrogating or supplementing a computationally burdensome portion of each case study and embedding them into existing pipelines, the work presented in this dissertation offers a blueprint for building data-driven enhanced methods that remain both accurate and tractable as models and devices continue to grow in complexity.

Bibliography

[1]M. Alexis, G. Mnatsakanyan, and C. Thiele, "Quantum signal processing and nonlinear Fourier analysis", Revista Matemática Complutense **37**, 655–694 (2024).

[2]R. Anirudh, R. Archibald, M. S. Asif, M. M. Becker, S. Benkadda, P.-T. Bremer, R. H. S. Budé, C. S. Chang, L. Chen, R. M. Churchill, J. Citrin, J. A. Gaffney, A. Gainaru, W. Gekelman, T. Gibbs, S. Hamaguchi, C. Hill, K. Humbird, S. Jalas, S. Kawaguchi, G.-H. Kim, M. Kirchen, S. Klasky, J. L. Kline, K. Krushelnick, B. Kustowski, G. Lapenta, W. Li, T. Ma, N. J. Mason, A. Mesbah, C. Michoski, T. Munson, I. Murakami, H. N. Najm, K. E. J. Olofsson, S. Park, J. L. Peterson, M. Probst, D. Pugmire, B. Sammuli, K. Sawlani, A. Scheinker, D. P. Schissel, R. J. Shalloo, J. Shinagawa, J. Seong, B. K. Spears, J. Tennyson, J. Thiagarajan, C. M. Ticoş, J. Trieschmann, J. van Dijk, B. Van Essen, P. Ventzek, H. Wang, J. T. L. Wang, Z. Wang, K. Wende, X. Xu, H. Yamada, T. Yokoyama, and X. Zhang, "2022 review of data-driven plasma science", IEEE Transactions on Plasma Science **51**, 1750–1838 (2023).

[3]E. Apra, E. J. Bylaska, W. A. De Jong, N. Govind, K. Kowalski, T. P. Straatsma, M. Valiev, H. J. van Dam, Y. Alexeev, J. Anchell, et al., "Nwchem: past, present, and future", The Journal of chemical physics **152** (2020).

[4]H. Arbabi and I. Mezić, "Ergodic Theory, Dynamic Mode Decomposition, and Computation of Spectral Properties of the Koopman Operator", SIAM Journal on Applied Dynamical Systems **16**, 2096–2126 (2017).

[5]A. Baiardi and M. Reiher, "Large-Scale Quantum Dynamics with Matrix Product States", Journal of Chemical Theory and Computation **15**, 3481–3498 (2019).

[6]H. Bassi, Y. Shen, H. S. Bhat, and R. Van Beeumen, "From noisy observables to accurate ground-state energies: a quantum–classical signal subspace approach with denoising", APL Quantum **2**, 046103 (2025).

[7]H. Bassi, Y. Zhu, S. Liang, J. Yin, C. C. Reeves, V. Vlček, and C. Yang, "Learning nonlinear integral operators via recurrent neural networks and its application in solving integro-differential equations", Machine Learning with Applications **15**, 100524 (2024).

[8]H. Bassi, Y. Zhu, E. Ye, P. Ren, A. Dektor, H. S. Bhat, and C. Yang, "Srafte: super-resolution and future time extrapolation for time-dependent pdes", arXiv preprint arXiv:2509.12220 (2025).

[9]L. Baumann, "On landau damping coupled with relaxation for the vlasov-poisson-bgk system", Master's the-sis, Julius-Maximilians-Universität Würzburg (2021).

[10]B. N. Bhaskar, G. Tang, and B. Recht, "Atomic norm denoising with applications to line spectral estimation", IEEE Transactions on Signal Processing **61**, 5987–5999 (2013).

[11]H. S. Bhat, H. Bassi, K. Ranka, and C. M. Isborn, "Incorporating memory into propagation of 1-electron reduced density matrices", Journal of Mathematical Physics **66** (2025).

[12]H. S. Bhat, K. Collins, P. Gupta, and C. M. Isborn, "Dynamic learning of correlation potentials for a time-dependent kohn-sham system", in Learning for dynamics and control conference (PMLR, 2022), pp. 546–558.

[13]J. S. Binkley, J. A. Pople, and W. J. Hehre, "Self-consistent molecular orbital methods. 21. small split-valence basis sets for first-row elements", Journal of the American Chemical Society **102**, 939–947 (1980).

[14]M. Blomquist, S. R. West, A. L. Binswanger, and M. Theillard, "Stable nodal projection method on octree grids", Journal of Computational Physics **499**, 112695 (2024).

[15]V. Blum, R. Asahi, J. Autschbach, C. Bannwarth, G. Bihlmayer, S. Blügel, L. A. Burns, T. D. Crawford, W. Dawson, W. A. de Jong, et al., "Roadmap on methods and software for electronic structure based simulations in chemistry and materials", Electronic structure **6**, 042501 (2024).

[16]M. Bohner and O. Tunç, "Qualitative analysis of integro-differential equations with variable retardation.", Discrete & Continuous Dynamical Systems-Series B **27** (2022).

[17]B. Bordelon, J. Cotler, C. Pehlevan, and J. A. Zavatone-Veth, "Dynamically learning to integrate in recurrent neural networks", arXiv preprint arXiv:2503.18754 (2025).

[18]S. L. Brunton and J. N. Kutz, *Data-driven science and engineering: machine learning, dynamical systems, and control* (Cambridge University Press, 2022).

[19]S. L. Brunton and J. N. Kutz, "Machine learning for partial differential equations", arXiv preprint arXiv:2303.17078 (2023).

[20]S. L. Brunton and J. N. Kutz, "Promising directions of machine learning for partial differential equations", Nature Computational Science **4**, 483–494 (2024).

[21]S. L. Brunton, J. L. Proctor, J. H. Tu, and J. N. Kutz, "Compressed sensing and dynamic mode decomposition", Journal of Computational Dynamics **2**, 165–191 (2016).

[22]S. L. Brunton, M. Budišić, E. Kaiser, and J. N. Kutz, "Modern Koopman Theory for Dynamical Systems", SIAM Review **64**, 229–340 (2022).

[23]J. Burch, R. Torbert, T. Phan, L.-J. Chen, T. Moore, R. Ergun, J. Eastwood, D. Gershman, P. Cassak, M. Argall, et al., "Electron-scale measurements of magnetic reconnection in space", Science **352**, aaf2939 (2016).

[24]D. Burgarth, P. Facchi, A. Hahn, M. Johnsson, and K. Yuasa, "Strong error bounds for Trotter and Strang-splittings and their implications for quantum chemistry", Physical Review Research **6**, 043155 (2024).

[25]E. F. Can, A. Ezen-Can, and F. Can, "Multilingual sentiment analysis: An RNN-based framework for limited data", arXiv preprint arXiv:1806.04511 (2018).

[26]W. Cao and W. Zhang, "Machine learning of partial differential equations from noise data", Theoretical and Applied Mechanics Letters **13**, 100480 (2023).

[27]Y. Cao, J. Romero, J. P. Olson, M. Degroote, P. D. Johnson, M. Kieferová, I. D. Kivlichan, T. Menke, B. Peropadre, N. P. Sawaya, et al., "Quantum chemistry in the age of quantum computing", Chemical Reviews **119**, 10856–10915 (2019).

[28]G. K. Chan, A. Keselman, N. Nakatani, Z. Li, and S. R. White, "Matrix product operators, matrix product states, and ab initio density matrix renormalization group algorithms", The Journal of chemical physics **145** (2016).

[29]G. K.-L. Chan and S. Sharma, "The density matrix renormalization group in quantum chemistry", Annual review of physical chemistry **62**, 465–481 (2011).

[30]A. Chattopadhyay and P. Hassanzadeh, "Long-term instabilities of deep learning-based digital twins of the climate system: the cause and a solution", arXiv preprint arXiv:2304.07029 (2023).

[31]R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural Ordinary Differential Equations", arXiv:1806.07366 [cs, stat] (2019).

[32]X. Chen, J. Duan, and G. E. Karniadakis, "Learning and meta-learning of stochastic advection–diffusion–reaction systems from sparse measurements", European Journal of Applied Mathematics **32**, 397–420 (2021).

[33]A. M. Childs, Y. Su, M. C. Tran, N. Wiebe, and S. Zhu, "Theory of trotter error with commutator scaling", Physical Review X **11**, 011020 (2021).

[34]K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation", arXiv preprint arXiv:1406.1078 (2014).

[35]R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, "Quantum algorithms revisited", Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences **454**, 339–354 (1998).

[36]G. Cohen and E. Rabani, "Memory effects in nonequilibrium quantum impurity models", Physical Review B **84**, 075150 (2011).

[37]M. Darman, P. Hassanzadeh, L. Zanna, and A. Chattopadhyay, "Fourier analysis of the physics of transfer learning for data-driven subgrid-scale models of ocean turbulence", arXiv preprint arXiv:2504.15487 (2025).

[38]S. T. Dawson, M. S. Hemati, M. O. Williams, and C. W. Rowley, "Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition", Experiments in Fluids **57**, 42 (2016).

[39]N. P. De Leon, K. M. Itoh, D. Kim, K. K. Mehta, T. E. Northup, H. Paik, B. Palmer, N. Samarth, S. Sangtawesin, and D. W. Steuerman, "Materials challenges and opportunities for quantum computing hardware", Science **372**, eabb2823 (2021).

[40]L. Debnath and L. Debnath, *Nonlinear partial differential equations for scientists and engineers* (Springer, 2005).

[41]Z. Ding, E. N. Epperly, L. Lin, and R. Zhang, "The esprit algorithm under high noise: Optimal error scaling and noisy super-resolution", in 2024 ieee 65th annual symposium on foundations of computer science (focs) (IEEE, 2024), pp. 2344–2366.

[42]Z. Ding, H. Li, L. Lin, H. Ni, L. Ying, and R. Zhang, "Quantum Multiple Eigenvalue Gaussian filtered search: an efficient and versatile quantum phase estimation method", Quantum **8**, 1487 (2024).

[43]Z. Ding and L. Lin, "Even shorter quantum circuit for phase estimation on early fault-tolerant quantum computers with applications to ground-state energy estimation", PRX Quantum **4**, 020331 (2023).

[44]Z. Ding and L. Lin, "Simultaneous estimation of multiple eigenvalues with short-depth quantum circuit on early fault-tolerant quantum computers", Quantum **7**, 1136 (2023).

[45]D. Du and E. Figueroa, "Learning spatially structured open quantum dynamics with regional-attention transformers", arXiv preprint arXiv:2509.06871 (2025).

[46]W. Dür, M. Hein, J. I. Cirac, and H.-J. Briegel, "Standard forms of noisy quantum operations via depolarization", Physical Review A **72**, 052326 (2005).

[47]K. Duraisamy, "Perspectives on machine learning-augmented reynolds-averaged and large eddy simulation models of turbulence", Physical Review Fluids **6**, 050504 (2021).

[48]W. D. Fries, X. He, and Y. Choi, "Lasdi: parametric latent space dynamics identification", Computer Methods in Applied Mechanics and Engineering **399**, 115436 (2022).

[49]K. Fukami and K. Taira, "Single-snapshot machine learning for super-resolution of turbulence", Journal of Fluid Mechanics **1001**, A32 (2024).

[50]H. Gao, X. Han, X. Fan, L. Sun, L.-P. Liu, L. Duan, and J.-X. Wang, "Bayesian conditional diffusion models for versatile spatiotemporal turbulence generation", Computer Methods in Applied Mechanics and Engineering **427**, 117023 (2024).

[51]H. Gao, L. Sun, and J.-X. Wang, "Super-resolution and denoising of fluid flow using physics-informed convolutional neural networks without high-resolution labels", Physics of Fluids **33** (2021).

[52]A. Georges, G. Kotliar, W. Krauth, and M. J. Rozenberg, "Dynamical mean-field theory of strongly correlated fermion systems and the limit of infinite dimensions", Reviews of modern physics **68**, 13 (1996).

[53]K. Georgopoulos, C. Emary, and P. Zuliani, "Modeling and simulating the noisy behavior of near-term quantum computers", Physical Review A **104**, 062432 (2021).

[54]J. C. Getelina, P. Sharma, T. Iadecola, P. P. Orth, and Y.-X. Yao, "Quantum subspace expansion in the presence of hardware noise", APL Quantum **1**, `https://doi.org/10.1063/5.0217294` (2024).

[55]P. Gibbon, "Introduction to plasma physics", arXiv preprint arXiv:2007.04783 (2020).

[56]S. S. Girimaji, "Turbulence closure modeling with machine learning: a foundational physics perspective", New Journal of Physics **26**, 071201 (2024).

[57]C. Gros and R. Valenti, "Cluster expansion for the self-energy: a simple many-body method for interpreting the photoemission spectra of correlated fermi systems", Physical Review B **48**, 418 (1993).

[58]J. Grosek and J. N. Kutz, *Dynamic Mode Decomposition for Real-Time Background/Foreground Separation in Video*, 2014.

[59]E. K. U. Gross and N. T. Maitra, "Introduction to TDDFT", in *Fundamentals of Time-Dependent Density Functional Theory*, edited by M. A. Marques, N. T. Maitra, F. M. Nogueira, E. Gross, and A. Rubio (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012), pp. 53–99.

[60]M. Großmann, M. Thieme, M. Grunert, and E. Runge, "Many-body perturbation theory vs. density functional theory: a systematic benchmark for band gaps of solids", arXiv preprint arXiv:2508.05247 (2025).

[61]J. Harlim, S. W. Jiang, S. Liang, and H. Yang, "Machine learning for prediction with missing dynamics", Journal of Computational Physics **428**, 109922 (2021).

[62]P. Hawinkel, E. Swinnen, S. Lhermitte, B. Verbist, J. Van Orshoven, and B. Muys, "A time series processing tool to extract climate-driven interannual vegetation dynamics using Ensemble Empirical Mode Decomposition (eemd)", Remote Sensing of Environment **169**, 375–389 (2015).

[63]X. He, Y. Choi, W. D. Fries, J. L. Belof, and J.-S. Chen, "Glasdi: parametric physics-informed greedy latent space dynamics identification", Journal of Computational Physics **489**, 112267 (2023).

[64]X. He, A. Tran, D. M. Bortz, and Y. Choi, "Physics-informed active learning with simultaneous weak-form latent space dynamics identification", International Journal for Numerical Methods in Engineering **126**, e7634 (2025).

[65]S. Hochreiter and J. Schmidhuber, "Long short-term memory", Neural computation **9**, 1735–1780 (1997).

[66]T. Holton, *Digital signal processing: principles and applications* (Cambridge University Press, Cambridge, UK, 2021).

[67]J. Hu, X. Wang, Y. Zhang, D. Zhang, M. Zhang, and J. Xue, "Time series prediction method based on variant LSTM recurrent neural network", Neural Processing Letters **52**, 1485–1500 (2020).

[68]C.-K. Ing, "Accumulated prediction errors, information criteria and optimal forecasting for autoregressive time series", The Annals of Statistics **35**, 1238–1277 (2007).

[69]A. Jacot, F. Gabriel, and C. Hongler, "Neural tangent kernel: convergence and generalization in neural networks", Advances in neural information processing systems **31** (2018).

[70]A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, et al., *Quantum computing with Qiskit*, 2024.

[71]J. Jia and A. R. Benson, "Neural Jump Stochastic Differential Equations", arXiv:1905.10403 [cs, stat] (2020).

[72]S. Johnstun and J.-F. Van Huele, "Understanding and compensating for noise on IBM quantum computers", American Journal of Physics **89**, 935–942 (2021).

[73]F. Ju, X. Wei, L. Huang, A. J. Jenkins, L. Xia, J. Zhang, J. Zhu, H. Yang, B. Shao, P. Dai, et al., "Acceleration without disruption: dft software as a service", Journal of Chemical Theory and Computation **20**, 10838–10851 (2024).

[74]L. P. Kadanoff, *Quantum statistical mechanics* (CRC Press, 2018).

[75]M. Kalimuthu, D. Holzmüller, and M. Niepert, "Loglo-fno: efficient learning of local and global features in fourier neural operators", arXiv preprint arXiv:2504.04260 (2025).

[76]M. Kamb, E. Kaiser, S. L. Brunton, and J. N. Kutz, "Time-delay observables for Koopman: Theory and applications", SIAM Journal on Applied Dynamical Systems **19**, 886–917 (2020).

[77]A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets", Nature **549**, 242–246 (2017).

[78]H. Kantz and T. Schreiber, *Nonlinear time series analysis*, 2nd (Cambridge University Press, Cambridge, UK, 2003).

[79]F. Karim, S. Majumdar, H. Darabi, and S. Chen, "LSTM fully convolutional networks for time series classification", IEEE access **6**, 1662–1669 (2017).

[80]J. Kaye and H. UR Strand, "A fast time domain solver for the equilibrium Dyson equation", Advances in Computational Mathematics **49**, 63 (2023).

[81]M. J. Kearns and U. V. Vazirani, *An introduction to computational learning theory* (MIT Press, Cambridge, MA, USA, 1994).

[82]I. G. Kevrekidis and G. Samaey, "Equation-free multiscale computation: Algorithms and applications", Annual Review of Physical Chemistry **60**, 321–344 (2009).

[83]S. Khodakarami, V. Oommen, A. Bora, and G. E. Karniadakis, "Mitigating spectral bias in neural operators via high-frequency scaling for physical systems", arXiv preprint arXiv:2503.13695 (2025).

[84]Y. Kim, Y. Choi, D. Widemann, and T. Zohdi, "A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder", Journal of Computational Physics **451**, 110841 (2022).

[85]D. Kingma and J. Ba, "Adam: a method for stochastic optimization", in International conference on learning representations (iclr) (2015).

[86]D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980 (2014).

[87]A. Y. Kitaev, *Quantum measurements and the Abelian Stabilizer problem*, 1995.

[88]K. Klymko, C. Mejuto-Zaera, S. J. Cotton, F. Wudarski, M. Urbanek, D. Hait, M. Head-Gordon, K. B. Whaley, J. Moussa, N. Wiebe, et al., "Real-time evolution for ultracompact Hamiltonian eigenstates on quantum hardware", PRX Quantum **3**, 020323 (2022).

[89]G. Kotliar, S. Y. Savrasov, K. Haule, V. S. Oudovenko, O. Parcollet, and C. Marianetti, "Electronic structure calculations with dynamical mean-field theory", Reviews of Modern Physics **78**, 865–951 (2006).

[90]N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Neural operator: learning maps between function spaces with applications to pdes", Journal of Machine Learning Research **24**, 1–97 (2023).

[91]J. N. Kutz, M. Reza, F. Faraji, and A. Knoll, "Shallow recurrent decoder for reduced order modeling of plasma dynamics", arXiv preprint arXiv:2405.11955 (2024).

[92]J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor, *Dynamic mode decomposition: data-driven modeling of complex systems* (SIAM, Philadelphia, PA, 2016).

[93]L. Lacombe and N. T. Maitra, "Developing new and understanding old approximations in TDDFT", Faraday Discussions **224**, 382–401 (2020).

[94]H. R. Larsson, H. Zhai, K. Gunst, and G. K.-L. Chan, "Matrix product states with large sites", Journal of Chemical Theory and Computation **18**, 749–762 (2022).

[95]H. R. Larsson, H. Zhai, C. J. Umrigar, and G. K.-L. Chan, "The chromium dimer: closing a chapter of quantum chemistry", Journal of the American Chemical Society **144**, 15932–15937 (2022).

[96]S. Lee, J. Lee, H. Zhai, Y. Tong, A. M. Dalzell, A. Kumar, P. Helms, J. Gray, Z.-H. Cui, W. Liu, et al., "Evaluating the evidence for exponential quantum advantage in ground-state quantum chemistry", Nature Communications **14**, 1952 (2023).

[97]H. Lei, P. Xie, L. Zhang, et al., "Machine learning-assisted multi-scale modeling", Journal of Mathematical Physics **64** (2023).

[98]P. Li, X. Liu, M. Chen, P. Lin, X. Ren, L. Lin, C. Yang, and L. He, "Large-scale ab initio simulations based on systematically improvable atomic basis", Computational Materials Science **112**, 503–517 (2016).

[99]W. Li, W. Liao, and A. Fannjiang, "Super-Eesolution Limit of the ESPRIT Algorithm", IEEE Transactions on Information Theory **66**, 4593–4608 (2020).

[100]X. Li, N. Govind, C. Isborn, A. E. DePrince III, and K. Lopata, "Real-time time-dependent electronic structure theory", Chemical Reviews **120**, 9951–9993 (2020).

[101]X. Li, T.-K. L. Wong, R. T. Chen, and D. K. Duvenaud, "Scalable gradients and variational inference for stochastic differential equations", in Symposium on advances in approximate bayesian inference (PMLR, 2020), pp. 1–28.

[102]Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations", in International conference on learning representations (2021).

[103]W. Liao and A. Fannjiang, "MUSIC for single-snapshot spectral estimation: Stability and super-resolution", Applied and Computational Harmonic Analysis **40**, 33–67 (2016).

[104]B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution", in Proceedings of the ieee Conference on Computer Vision and Pattern Recognition Workshops (2017), pp. 136–144.

[105]T. de Lima Silva, L. Borges, and L. Aolita, *Fourier-based quantum signal processing*, 2022.

[106]L. Lin, *Lecture Notes on Quantum Algorithms for Scientific Computation*, 2022.

[107]L. Lin and Y. Tong, "Near-optimal ground state preparation", Quantum **4**, 372 (2020).

[108]X. Liu, T. Xiao, S. Si, Q. Cao, S. Kumar, and C.-J. Hsieh, "Neural SDE: stabilizing Neural ODE Networks with Stochastic Noise", arXiv:1906.02355 [cs, stat] (2019).

[109]Z. Long, Y. Lu, X. Ma, and B. Dong, "PDE-net: learning PDEs from data", in International Conference on Machine Learning (PMLR, 2018), pp. 3208–3216.

[110]F. Lu, M. Maggioni, and S. Tang, "Learning interaction kernels in stochastic systems of interacting particles from multiple trajectories", Foundations of Computational Mathematics, 1–55 (2021).

[111]L. Lu, P. Jin, and G. E. Karniadakis, "DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators", arXiv preprint arXiv:1910.03193 (2019).

[112]L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, "Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators", Nature machine intelligence **3**, 218–229 (2021).

[113]G. Mahan, *Many-body physics*, 2000.

[114]N. T. Maitra, "Perspective: fundamental aspects of time-dependent density functional theory", The Journal of Chemical Physics **144**, 220901 (2016).

[115]S. Mallat, *A wavelet tour of signal processing* (Academic Press, Burlington, MA, 2009).

[116]A. Mauroy, Y. Susuki, and I. Mezic, *Koopman operator in systems and control*, Vol. 7 (Springer, 2020).

[117]M. McCabe, P. Harrington, S. Subramanian, and J. Brown, "Towards stability of autoregressive neural operators", arXiv preprint arXiv:2306.10619 (2023).

[118]A. J. McCaskey, Z. P. Parks, J. Jakowski, S. V. Moore, T. D. Morris, T. S. Humble, and R. C. Pooser, "Quantum chemistry as a benchmark for near-term quantum computers", npj Quantum Information **5**, 99 (2019).

[119]J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classical algorithms", New Journal of Physics **18**, 023023 (2016).

[120]R. McWeeny, *Methods of molecular quantum mechanics*, Second, Chemistry, Physical and Theoretical (Academic Press, 1989).

[121]X. Meng, Z. Li, D. Zhang, and G. E. Karniadakis, "PPINN: parareal physics-informed neural network for time-dependent PDEs", Computer Methods in Applied Mechanics and Engineering **370**, 113250 (2020).

[122]Y. Mi, P. Ren, H. Xu, H. Liu, Z. Wang, Y. Guo, J.-R. Wen, H. Sun, and Y. Liu, "Conservation-informed graph learning for spatiotemporal dynamics prediction", in Proceedings of the 31st acm sigkdd conference on knowledge discovery and data mining v. 1 (2025), pp. 1056–1067.

[123]K. Mizuta and T. Kuwahara, *Trotterization is substantially efficient for low-energy states*, 2025.

[124]D. Motlagh and N. Wiebe, "Generalized Quantum Signal Processing", PRX Quantum **5**, 020368 (2024).

[125]C. A. Myers, K. Miyazaki, T. Trepl, C. M. Isborn, and N. Ananth, "Gpu-accelerated on-the-fly nonadiabatic semiclassical dynamics", The Journal of Chemical Physics **161** (2024).

[126]Y. Ohmichi, Y. Sugioka, and K. Nakakita, "Stable dynamic mode decomposition algorithm for noisy pressure-sensitive-paint measurement data", AIAA Journal **60**, 1965–1970 (2022).

[127]P. J. Ollitrault, A. Kandala, C.-F. Chen, P. K. Barkoutsos, A. Mezzacapo, M. Pistoia, S. Sheldon, S. Woerner, J. M. Gambetta, and I. Tavernelli, "Quantum equation of motion for computing molecular excitation energies on a noisy quantum processor", Physical Review Research **2**, 043140 (2020).

[128]A. F. Oskooi, D. Roundy, M. Ibanescu, P. Bermel, J. D. Joannopoulos, and S. G. Johnson, "Meep: a flexible free-software package for electromagnetic simulations by the FDTD method", Computer Physics Communications **181**, 687–702 (2010).

[129]S. E. Otto and C. W. Rowley, "Koopman operators for estimation and control of dynamical systems", Annual Review of Control, Robotics, and Autonomous Systems **4**, 59–87 (2021).

[130]S. Pairault, D. Senechal, and A.-M. Tremblay, "Strong-coupling perturbation theory of the hubbard model", The European Physical Journal B-Condensed Matter and Complex Systems **16**, 85–105 (2000).

[131]S. Pairault, D. Sénéchal, and A.-M. Tremblay, "Strong-coupling expansion for the hubbard model", Physical review letters **80**, 5389 (1998).

[132]C.-Y. Park, "Efficient ground state preparation in variational quantum eigensolver with symmetry-breaking layers", APL Quantum **1**, https://doi.org/10.1063/5.0186205 (2024).

[133]M. Pasini, J. Nistal, S. Lattner, and G. Fazekas, "Continuous autoregressive models with noise augmentation avoid error accumulation", arXiv preprint arXiv:2411.18447 (2024).

[134]A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library", in *Advances in neural information processing systems 32* (Curran Associates, Inc., 2019), pp. 8024–8035.

[135]J. P. Perdew, "Density functional theory and the band gap problem", International Journal of Quantum Chemistry **28**, 497–523 (1985).

[136]E. Pollak, "A Tight Lower Bound to the Ground-State Energy", Journal of Chemical Theory and Computation **15**, 4079–4087 (2019).

[137]G. Prony, "Eassi expérimental et analytique: sur les lois de la dilatabilité des fluides élastiques et sur celles de la force expansive de la vapeur de l'eau et de la vapeur de l'alkool, a differentes températures", J. de l'Ecole Polytechnique **2**, 24–76 (1795).

[138]S. Qin, F. Lyu, W. Peng, D. Geng, J. Wang, X. Tang, S. Leroyer, N. Gao, X. Liu, and L. L. Wang, "Toward a Better Understanding of Fourier Neural Operators from a Spectral Perspective", arXiv:2404.07200 (2024).

[139]Y. Qin, J. Ma, M. Jiang, C. Dong, H. Fu, L. Wang, W. Cheng, and Y. Jin, "Data-driven modeling of landau damping by physics-informed neural networks", Physical Review Research **5**, 033079 (2023).

[140]A. A. Ramabathiran and P. Ramachandran, "SPINN: sparse, physics-based, and partially interpretable neural networks for PDEs", Journal of Computational Physics **445**, 110600 (2021).

[141]K. Ranka and C. M. Isborn, "Size-dependent errors in real-time electron density propagation", The Journal of Chemical Physics **158**, 174102 (2023).

[142]C. C. Reeves, Y. Zhu, C. Yang, and V. Vlcek, "On the unimportance of memory for the time non-local components of the Kadanoff-Baym equations", Phys. Rev. B **108**, 115152 (2023).

[143]C. C. Reeves, J. Yin, Y. Zhu, K. Z. Ibrahim, C. Yang, and V. Vl ček, "Dynamic mode decomposition for extrapolating nonequilibrium Green's-function dynamics", Phys. Rev. B **107**, 075107 (2023).

[144]M. Reiher, N. Wiebe, K. M. Svore, D. Wecker, and M. Troyer, "Elucidating reaction mechanisms on quantum computers", Proceedings of the National Academy of Sciences **114**, 7555–7560 (2017).

[145]L. Reining, "The GW approximation: content, successes and limitations", WIREs Computational Molecular Science **8**, e1344 (2018).

[146]P. Ren, N. B. Erichson, J. Guo, S. Subramanian, O. San, Z. Lukic, and M. W. Mahoney, "Superbench: a super-resolution benchmark dataset for scientific machine learning", Journal of Data-centric Machine Learning Research **8**, Dataset Certification, Reproducibility Certification, 1–45 (2025).

[147]P. Ren, C. Rao, Y. Liu, Z. Ma, Q. Wang, J.-X. Wang, and H. Sun, "Physr: physics-informed deep super-resolution for spatiotemporal data", Journal of Computational Physics **492**, 112438 (2023).

[148]S. Resch and U. R. Karpuzcu, "Benchmarking Quantum Computers and the Impact of Quantum Noise", ACM Computing Surveys (CSUR) **54**, 1–35 (2021).

[149]O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation", in International conference on medical Image Computing and Computer-Assisted Intervention (Springer, 2015), pp. 234–241.

[150]B. O. Roos, "The ground state potential for the chromium dimer revisited", Collection of Czechoslovak Chemical Communications **68**, 265–274 (2003).

[151]R. Roy and T. Kailath, "Esprit-estimation of signal parameters via rotational invariance techniques", IEEE Transactions on Acoustics, Speech, and Signal Processing **37**, 984–995 (1989).

[152]C. Salvi, M. Lemercier, and A. Gerasimovics, *Neural stochastic partial differential equations*, 2021.

[153]G. Samaey, W. Vanroose, D. Roose, and I. G. Kevrekidis, "Coarse-grained computation of traveling waves of lattice boltzmann models with newton-krylov solvers", arXiv preprint physics/0604147 (2006).

[154]P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data", Journal of Fluid Mechanics **656**, 5–28 (2010).

[155]P. J. Schmid, L. Li, M. P. Juniper, and O. Pust, "Applications of the dynamic mode decomposition", Theoretical and computational fluid dynamics **25**, 249–259 (2011).

[156]R. Schmidt, "Multiple emitter location and signal parameter estimation", IEEE Transactions on Antennas and Propagation **34**, 276–280 (1986).

[157]C. N. Self, K. E. Khosla, A. W. Smith, F. Sauvage, P. D. Haynes, J. Knolle, F. Mintert, and M. Kim, "Variational quantum algorithm with information sharing", npj Quantum Information **7**, 116 (2021).

[158]M. El-Shafey, "A new method for estimating complex exponentials to fit signals using samples of the signal and its derivative (or integral)", International Journal of Systems Science **26**, 1729–1737 (1995).

[159]S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: from theory to algorithms* (Cambridge University Press, USA, 2014).

[160]Y. Shen, A. Buzali, H.-Y. Hu, K. Klymko, D. Camps, S. F. Yelin, and R. V. Beeumen, *Efficient Measurement-Driven Eigenenergy Estimation with Classical Shadows*, 2024.

[161]Y. Shen, D. Camps, A. Szasz, S. Darbha, K. Klymko, D. B. Williams–Young, N. M. Tubman, and R. V. Beeumen, "Estimating Eigenenergies from Quantum Dynamics: A Unified Noise-Resilient Measurement-Driven Approach", Quantum **9**, 1836 (2025).

[162]Y. Shen, K. Klymko, J. Sud, D. B. Williams-Young, W. A. de Jong, and N. M. Tubman, "Real-time Krylov Theory for Quantum Computing Algorithms", Quantum **7**, 1066 (2023).

[163]W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network", in Proceedings of the ieee conference on computer vision and pattern recognition (2016), pp. 1874–1883.

[164]S. Smidstrup, D. Stradi, J. Wellendorff, P. A. Khomyakov, U. G. Vej-Hansen, M.-E. Lee, T. Ghosh, E. Jónsson, H. Jónsson, and K. Stokbro, "First-principles green's-function method for surface calculations: a pseudopotential localized basis set approach", Physical Review B **96**, 195309 (2017).

[165]A. Smirne and B. Vacchini, "Nakajima-Zwanzig versus time-convolutionless master equation for the non-Markovian dynamics of a two-level system", Physical Review A **82**, 022110 (2010).

[166]J. Song, Z. Song, P. Ren, N. B. Erichson, M. W. Mahoney, and X. S. Li, "Forecasting high-dimensional spatio-temporal systems from sparse measurements", Machine Learning: Science and Technology **5**, 045067 (2024).

[167]T. D. Stanescu and G. Kotliar, "Strong coupling theory for interacting lattice models", Physical Review B—Condensed Matter and Materials Physics **70**, 205112 (2004).

[168]G. Stefanucci and R. van Leeuwen, *Nonequilibrium many-body theory of quantum systems: a modern introduction* (Cambridge University Press, 2013).

[169]I. Steinbach, "Phase-field model for microstructure evolution at the mesoscopic scale", Annual Review of Materials Research **43**, 89–107 (2013).

[170]P. Stoica and R. L. Moses, *Spectral analysis of signals* (Pearson Prentice Hall, Upper Saddle River, NJ, 2005).

[171]A. Subel, Y. Guan, A. Chattopadhyay, and P. Hassanzadeh, "Explaining the physics of transfer learning in data-driven turbulence modeling", PNAS nexus **2**, pgad015 (2023).

[172]L. Sun, H. Gao, S. Pan, and J.-X. Wang, "Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data", Computer Methods in Applied Mechanics and Engineering **361**, 112732 (2020).

[173]A. Szasz, E. Younis, and W. A. de Jong, "Ground state energy and magnetization curve of a frustrated magnetic system from real-time evolution on a digital quantum processor", Quantum **9**, 1704 (2025).

[174]F. Takens, "Detecting strange attractors in turbulence", in Dynamical systems and turbulence, warwick 1980. lecture notes in mathematics, vol 898. springer, berlin, heidelberg (Springer, 1981), pp. 366–381.

[175]M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains", Advances in neural information processing systems **33**, 7537–7547 (2020).

[176]Y. Tang, L. Qi, F. Xie, X. Li, C. Ma, and M.-H. Yang, "Predformer: transformers are effective spatial-temporal predictive learners", (2024).

[177]J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, et al., "The Variational Quantum Eigensolver: a review of methods and best practices", Physics Reports **986**, 1–128 (2022).

[178]N. M. Tubman, J. Lee, T. Y. Takeshita, M. Head-Gordon, and K. B. Whaley, "A deterministic alternative to the full configuration interaction quantum Monte Carlo method", The Journal of Chemical Physics **145**, https://doi.org/10.1063/1.4955109 (2016).

[179]R. Van Beeumen, K. Meerbergen, and W. Michiels, "Compact rational krylov methods for nonlinear eigenvalue problems", SIAM Journal on Matrix Analysis and Applications **36**, 820–838 (2015).

[180]M. J. Van Setten, F. Caruso, S. Sharifzadeh, X. Ren, M. Scheffler, F. Liu, J. Lischner, L. Lin, J. R. Deslippe, S. G. Louie, et al., "Gw 100: benchmarking g 0 w 0 for molecular systems", Journal of chemical theory and computation **11**, 5665–5687 (2015).

[181]D. Venturi and G. E. Karniadakis, "Convolutionless Nakajima–Zwanzig equations for stochastic analysis in nonlinear dynamical systems", Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **470**, 20130754 (2014).

[182]L. O. Wagner, Z.-h. Yang, and K. Burke, "Exact conditions and their relevance in tddft", Fundamentals of Time-Dependent Density Functional Theory, 101–123 (2012).

[183]X. Wang, L. Xiong, X. Cai, and X. Yuan, *Computing n-time correlation functions without ancilla qubits*, 2025.

[184]Z. Wang, J. Chen, and S. C. Hoi, "Deep learning for image super-resolution: a survey", IEEE Transactions on Pattern Analysis and Machine Intelligence **43**, 3365–3387 (2020).

[185]T. T. Warner, *Numerical weather and climate prediction* (Cambridge University Press, Cambridge, 2011).

[186]P.-Å. Wedin, "Perturbation theory for pseudo-inverses", BIT Numerical Mathematics **13**, 217–232 (1973).

[187]F. Weigend and R. Ahlrichs, "Balanced basis sets of split valence, triple zeta valence and quadruple zeta valence quality for h to rn: design and assessment of accuracy", Physical Chemistry Chemical Physics **7**, 3297–3305 (2005).

[188]E. Weinan, *Principles of multiscale modeling* (Cambridge University Press, 2011).

[189]E. Weinan and B. Engquist, "The heterognous multiscale methods", Communications in Mathematical Sciences **1**, 87–132 (2003).

[190]P. Wesseling, *Introduction to multigrid methods*, tech. rep. (1995).

[191]G. B. Whitham, "Variational methods and applications to water waves", Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences **299**, 6–25 (1967).

[192]D. B. Williams-Young, A. Petrone, S. Sun, T. F. Stetina, P. Lestrange, C. E. Hoyer, D. R. Nascimento, L. Koulias, A. Wildman, J. Kasper, et al., "The chronus quantum software package", Wiley Interdisciplinary Reviews: Computational Molecular Science **10**, e1436 (2020).

[193]S. Wun Cheung, Y. Choi, J.-L. Fattebert, and D. Osei-Kuffuor, "Model order reduction for quantum molecular dynamics", arXiv e-prints, arXiv–2509 (2025).

[194]P. Xie, "Ab initio multi-scale modeling of crystals: methods and applications in ferroelectrics", PhD thesis (Princeton University, 2024).

[195]P. Xie and W. E, "Coarse-graining conformational dynamics with multidimensional generalized langevin equation: how, when, and why", Journal of Chemical Theory and Computation **20**, 7708–7715 (2024).

[196]W. Xu, R. T. Q. Chen, X. Li, and D. Duvenaud, "Infinitely Deep Bayesian Neural Networks with Stochastic Differential Equations", arXiv:2102.06559 [cs, stat] (2021).

[197]E. Ye, *Reducing computational costs for many-body physics problems* (California Institute of Technology, 2021).

[198]E. Ye and N. F. Loureiro, "Quantum-inspired method for solving the Vlasov-Poisson equations", Physical Review E **106**, 035208 (2022).

[199]J. Yin, Y.-h. Chan, F. da Jornada, D. Qiu, C. Yang, and S. G. Louie, "Analyzing and predicting non-equilibrium many-body dynamics via dynamic mode decomposition", J. Comput. Phys. **477**, 111909 (2023).

[200]J. Yin, Y.-h. Chan, F. H. da Jornada, D. Y. Qiu, S. G. Louie, and C. Yang, "Using dynamic mode decomposition to predict the dynamics of a two-time non-equilibrium Green's function", J. Comput. Sci. **64**, 101843 (2022).

[201]J. Yin, Y.-H. Chan, F. H. da Jornada, D. Y. Qiu, C. Yang, and S. G. Louie, "Analyzing and predicting non-equilibrium many-body dynamics via dynamic mode decomposition", Journal of Computational Physics **477**, 111909 (2023).

[202]Y. S. Yordanov, D. R. Arvidsson-Shukur, and C. H. Barnes, "Efficient quantum circuits for quantum computational chemistry", Physical Review A **102**, 062612 (2020).

[203]D. Zhang, L. Lu, L. Guo, and G. E. Karniadakis, "Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems", Journal of Computational Physics **397**, 108850 (2019).

[204]J. Zhang and K.-F. Man, "Time series prediction using RNN in multi-dimension embedding phase space", in Smc'98 conference proceedings. 1998 ieee international conference on systems, man, and cybernetics (cat. no. 98ch36218), Vol. 2 (IEEE, 1998), pp. 1868–1873.

[205]S. Zhang, J. Carlson, and J. E. Gubernatis, "Constrained path monte carlo method for fermion ground states", Physical Review B **55**, 7464 (1997).

[206]S. Zhang and H. Krakauer, "Quantum monte carlo method using phase-free random walks with slater determinants", Physical review letters **90**, 136401 (2003).

[207]L. Zhao, J. Goings, K. Shin, W. Kyoung, J. I. Fuks, J.-K. Kevin Rhee, Y. M. Rhee, K. Wright, J. Nguyen, J. Kim, et al., "Orbital-optimized pair-correlated electron simulations on trapped-ion quantum computers", npj Quantum Information **9**, 60 (2023).

[208]Y. Zhu, "Mori-zwanzig equation: theory and applications", PhD thesis (University of California, Santa Cruz, 2019).

[209]Y. Zhu, J. M. Dominy, and D. Venturi, "On the estimation of the Mori-Zwanzig memory integral", Journal of Mathematical Physics **59** (2018).

[210]Y. Zhu and H. Lei, "Effective Mori-Zwanzig equation for the reduced-order modeling of stochastic systems", Discrete & Continuous Dynamical Systems - S (2021).

[211]Y. Zhu and H. Lei, "Effective Mori-Zwanzig equation for the reduced-order modeling of stochastic systems", Discrete and Continuous Dynamical Systems - S **15**, 959–982 (2022).

[212]Y. Zhu, P. Rosenberg, Z. Huang, H. Bassi, C. Yang, and S. Zhang, "Sigma-attention: a transformer-based operator learning framework for self-energy in strongly correlated systems", arXiv preprint arXiv:2504.14483 (2025).

[213]Y. Zhu, Y.-H. Tang, and C. Kim, "Learning stochastic dynamics with statistics-informed neural network", Journal of Computational Physics **474**, 111819 (2023).

[214]Y. Zhu and D. Venturi, "Generalized Langevin equations for systems with local interactions", Journal of Statistical Physics **178**, 1217–1247 (2020).

[215]Y. Zhu, J. Yin, C. C. Reeves, C. Yang, and V. Vlček, "Predicting nonequilibrium green's function dynamics and photoemission spectra via nonlinear integral operator learning", Machine Learning: Science and Technology **6**, 015027 (2025).

[216]S. Zhuk, N. F. Robertson, and S. Bravyi, "Trotter error bounds and dynamic multi-product formulas for Hamiltonian simulation", Physical Review Research **6**, 033309 (2024).

[217]R. Zwanzig, *Nonequilibrium statistical mechanics* (Oxford university press, 2001).

Appendix A

Appendix for Chapter 3

A.1 Intuition on approximate space-time separability and memory

SRaFTE couples a temporal update on the coarse grid with an instantaneous learned spatial lift to the fine grid. This architecture is most effective when a short-memory condition holds: conditioned on the present coarse field $u_c(t)$, the subgrid content is well approximated by a function of $u_c(t)$ alone,

$$u_f(t) \approx \Phi\Big(u_c(t)\Big),$$

so the one-step surrogate propagator $\hat{\mathcal{P}}_f(\Delta t;\theta) = \mathcal{N}_\theta \circ \mathcal{P}_c(\Delta t) \circ P$ is a reasonable ansatz for the fine solver. In that sense, SRaFTE exploits an approximate separation of temporal and spatial correlation: the coarse solver $\mathcal{P}_c$ carries the time advance while $\mathcal{N}_\theta$ provides an instantaneous spatial correction.

Related assumptions are also made intuitive in the SHRED [91] method, which reconstructs the full field from a short temporal history via a shallow recurrent encoder and a nonlinear decoder. In our notation, its closure has the form $u_f(t) \approx \Phi_h\Big(u_c(t-\tau{:}t)\Big)$, i.e., a history–dependent (non-Markovian) map. By contrast, SRaFTE delegates temporal evolution to the coarse integrator $\mathcal{P}_c$ and applies a time-local lift $\mathcal{N}_\theta$, corresponding to a Markovian closure $u_f(t) \approx \mathcal{N}_\theta\Big(u_c(t)\Big)$. From a separation of variables perspective, SHRED learns a nonlinear space–time factorization with an explicit recurrent temporal component, whereas SRaFTE composes the physics-based temporal factor $\mathcal{P}_c$ with an instantaneous spatial factor $\mathcal{N}_\theta$.

Table A.1: Core notation used in Chapter 3.

Symbol	**Meaning**
Ω	Physical spatial domain (continuous)
Ω_c, Ω_f	Coarse and fine computational grids/index sets discretizing Ω
X	State/function space for fields on Ω (e.g., $X = L^2(\Omega)$ or $L^1(\Omega)$)
$u_c(t)$, $u_f(t)$	Coarse and fine states at (continuous) time t
u_c^n, u_f^n	Coarse and fine states at discrete step n with step size Δt
P	Restriction/projection from fine to coarse (spatial decimation via point sampling)
$\mathcal{N}_\theta$	Learned lift / super-resolution operator (Phase 1 network)
$\mathcal{P}_f(\Delta t)$	Reference fine one-step propagator (ground truth solver)
$\mathcal{P}_c(\Delta t)$	Coarse numerical one-step propagator
$\widehat{\mathcal{P}}_f(\Delta t)$	Phase 2 surrogate fine one-step propagator *(*$\widehat{\mathcal{P}}_f(\Delta t) := \mathcal{N}_\theta \circ \mathcal{P}_c(\Delta t) \circ P$*)*
Φ_θ	Autoregressive (AR) baseline one-step map on the fine grid
Δt	Timestep
T	Number of time steps used for evaluation
N_c, N_f	Number of grid points on coarse and fine discretizations
K	Number of supervised training pairs
N_{traj}	Number of training trajectories
AR, SR	Shorthand for autoregressive and super-resolution settings

Linear intuition (heat / wave). For linear, constant-coefficient PDEs with periodic BCs (e.g., $\partial_t u = \nu\Delta u + \alpha f$ or $\partial_{tt} u = c^2\Delta u$), a Fourier projection onto low ($|k| \le k_c$) and high ($|k| > k_c$) modes commutes with the generator. Each wavenumber evolves independently in time:

$$\begin{aligned} \widehat{u}_k(t) &= e^{\lambda_k t}\,\widehat{u}_k(0) + (\text{forced response}), \\ \lambda_k &= \begin{cases} -\nu|k|^2 & \text{(heat)}, \\ \pm i\,c|k| & \text{(wave)}. \end{cases} \end{aligned} \tag{A.1}$$

Thus, the *temporal* evolution is multiplicative and modewise, while the *spatial* content is set by which modes are present. For the heat equation, high modes decay on a time scale $\tau_k \sim (\nu|k|^2)^{-1}$; when k_c is chosen so that τ_{k_c} is significantly smaller than the resolved time scale, the unresolved content is quickly chained to the resolved part, justifying an instantaneous lift. For the wave equation, although the dynamics are nondissipative, the (k,ω) spectrum concentrates along the dispersion ridge $\omega = c|k|$, so phase is primarily governed by $\mathcal{P}_c$ and the lift mainly corrects spatial aliasing.

Nonlinear intuition (2D NSE) Writing $\omega = \omega_< + \omega_>$ for low/high modes, we take $\mathbf{P}_<$ and $\mathbf{P}_>$ to be orthogonal Fourier projectors at a cutoff k_c: for any solution on a grid $f(x) = \sum_{k\in\mathbb{Z}^2} \hat{f}_k e^{ikx}$,

$$\mathbf{P}_< f := \sum_{|k|\le k_c} \hat{f}_k e^{ik\cdot x}, \qquad \mathbf{P}_> f := f - \mathbf{P}_< f,$$

so that $\mathbf{P}_<^2 = \mathbf{P}_<$, $\mathbf{P}_>^2 = \mathbf{P}_>$, and $\mathbf{P}_<\mathbf{P}_> = 0$. Both of these operators reduce a fine grid solution to its coarse counterpart, but the earlier spatial decimation projector P is a practical grid-level downsampler (not necessarily orthogonal or spectral) used to form $u_c = Pu_f$, whereas the present projectors $\mathbf{P}_<$ and $\mathbf{P}_>$ are the ideal orthogonal Fourier projector at a cutoff k_c (these are used only in this subsection for the sake of analysis). This can coincide with P only when P implements spectral truncation in conjunction with spatial decimation. Using these operators yields

$$\begin{aligned} \partial_t \omega_> = \nu\,\Delta\omega_> &+ \mathbf{P}_> B(\omega_<,\omega_<) + \mathbf{P}_> B(\omega_<,\omega_>) \\ &+ \mathbf{P}_> B(\omega_>,\omega_>). \end{aligned}$$

where B is the advection operator in 2D vorticity form, and the corresponding Mori–Zwanzig identity [209, 208, 210] expresses the resolved dynamics as

$$\partial_t \omega_< = \mathbf{P}_<\mathcal{L}\omega_< + \int_0^t K(t,s;\omega_<(s))\,ds + \eta(t),$$

with a memory kernel K and noise η. When viscosity and the cutoff k_c yield a spectral gap, high modes decay on a fast time $\tau_{\text{fast}} \sim (\nu k_c^2)^{-1}$ and the memory integral is well-approximated by a Markovian functional, $\int_0^t K(\cdot)\,ds \approx \Phi(\omega_<(t))$. SRaFTE's Phase 1 learns an analogous operator to Φ from the dataset (u_c, u_f); Phase 2 wraps it around the coarse integrator so that the *time* advance is done by $\mathcal{P}_c$ while the spatial correction is refreshed each step by $\mathcal{N}_\theta$.

Thus, the short-memory condition is more likely to be satisfied for (i) diffusive systems and (ii) linear dispersive systems with constant coefficients, as well as (iii) moderately nonlinear, viscous flows where high modes relax quickly relative to resolved dynamics. The short-memory condition can degrade (and Phase 2 may drift) when (a) the Reynolds number is large enough such that subgrid time scales are not fast compared to resolved scales, (b) strong nonlocal interactions create long memory, or (c) the downsampling ratio is so aggressive that crucial interaction bands move entirely into the unresolved set.

A.2 One–step test error generalization

Let $\mathcal{D}$ denote the distribution of fine snapshots u used in Phase 2. Define the population risk

$$\mathcal{R}(\theta) := \mathbb{E}_{u\sim\mathcal{D}} \Big\| \mathcal{P}_f(\Delta t)\, u - \mathcal{N}_\theta\Big[\mathcal{P}_c(\Delta t)\, Pu\Big] \Big\|. \tag{A.2}$$

Let $\widehat{\mathcal{R}}_m(\theta)$ be the empirical counterpart computed on m approximately independent per-step pairs.

Write the loss on one instance as $f_\theta(u) := \Big\| \mathcal{P}_f(\Delta t)u - \mathcal{N}_\theta\Big[\mathcal{P}_c(\Delta t)Pu\Big] \Big\| \in [0, B]$ and define the function class $\mathcal{F} := \{f_\theta : \theta \in \Theta\}$. Here Θ is the space of all feasible parameter vectors θ. Given a sample $S = (u^{(1)}, \ldots, u^{(m)})$ drawn i.i.d. from $\mathcal{D}$, denote the empirical mean $\widehat{\mathbb{E}}_S[f] = \frac{1}{m}\sum_{i=1}^m f(u^{(i)})$ and the population mean $\mathbb{E}[f] = \mathbb{E}_{u\sim\mathcal{D}}[f(u)]$. Define the empirical Rademacher complexity

$$\widehat{\mathfrak{R}}_m(\mathcal{F}) := \mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}} \frac{1}{m}\sum_{i=1}^m \sigma_i f(u^{(i)})\right]$$
$$\sigma_i \overset{\text{i.i.d.}}{\sim} \text{Unif}\{-1, +1\},$$

and, for brevity, write $\mathfrak{R}_m := \widehat{\mathfrak{R}}_m(\mathcal{F})$. Note that $\mathfrak{R}_m$ is a data–dependent measure of how well functions in $\mathcal{F}_\theta$ can fit random ± 1 noise. While such complexity measures are typically used to derive generalization bounds for i.i.d. classification problems, in the SRaFTE setting—where we learn continuous-valued PDE dynamics from temporally correlated trajectories——$\mathfrak{R}_m$ should be viewed only as a qualitative proxy for the capacity of the SRaFTE hypothesis class, rather than as a fully rigorous complexity measure.

Suppose the loss f_θ is bounded by $B > 0$. As we detail below, standard empirical risk minimization (via symmetrization and contraction) implies that with probability at least $1-\delta$,

$$\mathcal{R}(\theta) \le \widehat{\mathcal{R}}_m(\theta) + 2\mathfrak{R}_m + 3B\sqrt{\frac{\ln(2/\delta)}{2m}} \tag{A.3}$$

This bound is stated for independently, identically distributed (i.i.d.) one-step pairs (u^n, u^{n+1}); in practice, we subsample sequential steps from trajectories, thus the reported empirical results should be viewed as supportive evidence rather than a strict proof where the data contains temporal dependence. Therefore, the per–step test error is controlled by the observed Phase 2 training error plus a capacity/finite–sample term decaying as $m^{-1/2}$.

Derivation. *Uniform symmetrization.* Consider the deviation functional $\Phi(S) := \sup_{f\in\mathcal{F}}\Big(\mathbb{E}[f] - \widehat{\mathbb{E}}_S[f]\Big)$. Introducing a ghost sample $S' = (u'^{(1)}, \dots, u'^{(m)})$ i.i.d. as S and using standard symmetrization yields,

$$\begin{aligned}
\mathbb{E}_S[\Phi(S)] &= \mathbb{E}_S\Big[\sup_{f\in\mathcal{F}}\Big(\mathbb{E}_{u'}[f(u')] - \widehat{\mathbb{E}}_S[f]\Big)\Big] \\
&\le \mathbb{E}_{S,S'}\Big[\sup_{f\in\mathcal{F}}\Big(\widehat{\mathbb{E}}_{S'}[f] - \widehat{\mathbb{E}}_S[f]\Big)\Big] \\
&= \mathbb{E}_{S,S',\boldsymbol{\sigma}}\Big[\sup_{f\in\mathcal{F}}\frac{1}{m}\sum_{i=1}^m \sigma_i\Big(f(u'^{(i)}) - f(u^{(i)})\Big)\Big] \\
&\le 2\,\mathbb{E}_{S,\boldsymbol{\sigma}}\Big[\sup_{f\in\mathcal{F}}\frac{1}{m}\sum_{i=1}^m \sigma_i f(u^{(i)})\Big] = 2\mathfrak{R}_m.
\end{aligned}$$

Bounded differences concerntration. By McDiarmid's inequality, with probability at least $1-\delta$,

$$\Phi(S) \le \mathbb{E}_S[\Phi(S)] + B\sqrt{\frac{\ln(2/\delta)}{2m}}.$$

Combining with the symmetrization argument yields

$$\sup_{f\in\mathcal{F}}\Big(\mathbb{E}[f] - \widehat{\mathbb{E}}_S[f]\Big) \le 2\mathfrak{R}_m + B\sqrt{\frac{\ln(2/\delta)}{2m}} \quad \text{w.p. } \ge 1-\delta.$$

Applying the same bounded-differences argument to the random functional $S \mapsto \mathfrak{R}_m = \widehat{\mathfrak{R}}_m(\mathcal{F})$ and a union bound yields, with probability at least $1-\delta$,

$$\forall\,\theta\in\Theta: \quad \mathcal{R}(\theta) - \widehat{\mathcal{R}}_m(\theta) \le 2\mathfrak{R}_m + 3B\sqrt{\frac{\ln(2/\delta)}{2m}},$$

which is exactly (A.3).

Finite sample behavior of the test risk. To empirically validate the generalization prediction in (A.3) from multiple test trajectories, we randomly subsample m one-step pairs (u^n, u^{n+1}), evaluate $\|u^{n+1} - \widehat{u}^{n+1}\|$, average over the m samples, and plot the mean against $1/\sqrt{m}$. A near linear increase in Figure A.1 versus $1/\sqrt{m}$ matches the $O(m^{-1/2})$ finite-sample result in the generalization bound (A.3). Although per-step samples along a trajectory are dependent, the observed near-linear trend versus $1/\sqrt{m}$ aligns with the i.i.d. prediction and is robust under random subsampling. As expected, we find that larger m reduces variance in the empirical per-step risk.

A.3 Model summaries

Unless otherwise stated, models operate on stacks of coarse/fine grid solutions in time treated as channels and use `GELU` activations. Specifically, all models except FNO-AR take in coarse grid inputs and and all models output fine grid outputs.

Common building blocks

Spectral convolution (2D). All Fourier layers compute a real-space FFT over spatial dimensions (H, W), retain only a low-frequency rectangle of size $(m_1 \times m_2)$, apply a learned complex linear mixing on the retained modes, zero out all others, and return to the spatial domain by inverse FFT. Concretely, if $x \in \mathbb{C}^{B \times C_{\text{in}} \times H \times (W/2+1)}$ denotes the FFT, the retained block and the learned mixing is given by weights $W \in \mathbb{C}^{C_{\text{in}} \times C_{\text{out}} \times m_1 \times m_2}$, with

$$\widehat{y}_{b,o,:,:} = \sum_{i=1}^{C_{\text{in}}} \widehat{x}_{b,i,:,:} \odot W_{i,o,:,:}, \qquad y = \text{iFFT}(\widehat{y}).$$

A parallel 1×1 convolutional path is often summed with the spectral branch before a nonlinearity.

PixelShuffle upsampling. For scale factor $s \in \{2, 4, 8\}$ we map $C \mapsto s^2 C$ channels by a 3×3 convolution followed by `PixelShuffle`(s) and a GELU.

FUnet specification

- **Input and output.** The network maps a coarse input $x \in \mathbb{R}^{B \times C_{\text{in}} \times H_c \times W_c}$ to a fine output $y \in \mathbb{R}^{B \times C_{\text{out}} \times H_f \times W_f}$, with $C_{\text{in}} = C_{\text{out}} = 100$. The spatial resolutions satisfy

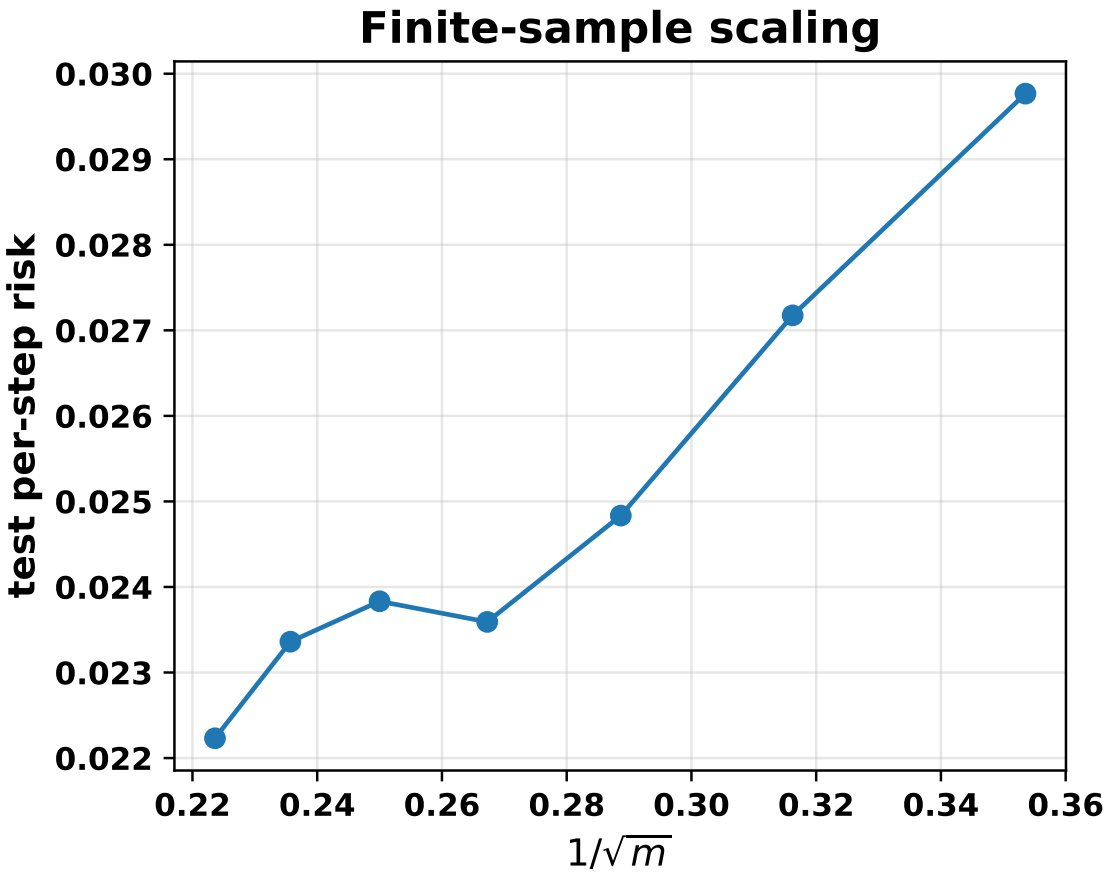


Figure A.1: **Finite sample scaling of the Phase 2 per–step test risk.** For random subsamples of size m from the test set, we compute the empirical per–step risk $\widehat{\mathcal{R}}_m(\theta) = \frac{1}{m}\sum_{i=1}^{m} \|\mathcal{P}_f(\Delta t)u^{(i)} - \mathcal{N}_\theta[\mathcal{P}_c(\Delta t)Pu^{(i)}]\|$ and plot it against $1/\sqrt{m}$. The nearly linear trend is consistent with the generalization bound in (A.3), which predicts a finite sample decay as $m^{-1/2}$.

$(H_f, W_f) = s(H_c, W_c)$ for an upsampling factor s (e.g., $s = 4$ in the $32\times32 \to 128\times128$ experiments).

- **Lifting layer.** A 1×1 convolution maps the 100 input channels to $C_{\text{lift}} = 128$ channels.
- **Encoder.** Two encoder stages operate at 128 channels. Each stage is a `ConvBlock` consisting of two consecutive 3×3 convolutions with GELU activations. After the second `ConvBlock`, a 2×2 max-pooling layer with stride 2 halves the spatial

resolution, and we store skip connections from both encoder stages.

- **Bottleneck.** The bottleneck first applies a `ConvBlock` at 256 channels, followed by a Fourier layer that performs a spectral convolution with retained modes $(m_1, m_2) = (32, 32)$ at 256 channels, and a final GELU activation.

- **Decoder.** The decoder consists of three upsampling stages based on pixel-shuffle operators with skip connections: starting from the bottleneck representation (256 channels), successive stages (1) reduce channels from 256 to 128 after combining with the deeper encoder skip, (2) reduce channels from 128 to 64 after combining with the shallower encoder skip, and (3) reduce channels from 64 to 32 on the finest grid. The product of the pixel-shuffle factors over the three stages equals the overall upsampling factor s.

- **Projection (head).** A 3×3 convolution maps 32 channels to 32 channels, followed by GELU and a final 3×3 convolution mapping 32 channels to $C_{\text{out}} = 100$.

- **Nonlinearity and normalization.** GELU activations are used throughout; no normalization layers are applied.

U-Net [149] specification

- **Input and output.** The U-Net maps $x \in \mathbb{R}^{B\times C_{\text{in}}\times H_c\times W_c}$ to $y \in \mathbb{R}^{B\times C_{\text{out}}\times H_f\times W_f}$, with $C_{\text{in}} = C_{\text{out}} = 100$ and $(H_f, W_f) = s(H_c, W_c)$.

- **Base width.** The base number of channels is $C_{\text{base}} = 64$.

- **Encoder.** There are two encoder levels. At level 1, two `conv_block`s operate at 64 channels, followed by a 2×2 max-pooling layer (stride 2). At level 2, two `conv_block`s operate at 128 channels, again followed by max-pooling. Each `conv_block` consists of a 3×3 convolution, batch normalization, and GELU activation. Skip connections are taken from the outputs of the encoder levels before pooling.

- **Bottleneck.** Two `conv_block`s operate at 256 channels on the most downsampled resolution.

- **Decoder.** Each decoder level upsamples via a `ConvTranspose2d` layer (stride 2), concatenates the corresponding encoder skip feature map, and applies two `conv_block`s. Channel counts mirror the encoder in reverse (256→128, then 128→64).

- **Projection (head).** A final 1×1 convolution maps 64 channels to $C_{\text{out}} = 100$. The output is then upsampled to the fine resolution (H_f, W_f) by bilinear interpolation when needed to match the desired scale factor s.
- **Nonlinearity and normalization.** All convolutions inside `conv_blocks` use GELU activations and batch normalization.

EDSR [104] specification

- **Input and output.** The EDSR network maps $x \in \mathbb{R}^{B\times C_{\text{in}}\times H_c\times W_c}$ to $y \in \mathbb{R}^{B\times C_{\text{out}}\times H_f\times W_f}$, with $C_{\text{in}} = C_{\text{out}} = 100$.
- **Width and depth.** We use `n_feats` $= 128$ feature channels and `n_res_blocks` $= 16$ residual blocks.
- **Head.** A 3×3 convolution maps $C_{\text{in}} = 100$ channels to 128 channels.
- **Body.** The body consists of 16 `ResBlocks` in series. Each `ResBlock` applies a 3×3 convolution, ReLU activation, and a second 3×3 convolution, with a residual connection scaled by `res_scale` $= 0.1$. A final 3×3 convolution at 128 channels is applied at the end of the body, and the head output is added as a skip connection (standard EDSR formulation).
- **Tail (upsampling and projection).** An `Upsampler` implements the overall spatial upsampling factor s (for example, two pixel-shuffle layers with factor 2 each when $s = 4$). A concluding 3×3 convolution maps 128 channels back to $C_{\text{out}} = 100$.
- **Input normalization.** Before the head, we apply channelwise affine normalization using stored mean and standard deviation; the inverse transform is applied after the tail.
- **Nonlinearity and normalization.** ReLU activations are used in the residual blocks; no batch normalization is used inside residual blocks (as in the original EDSR).

FNO-SR [102] specification

- **Input and output.** FNO-SR maps $x \in \mathbb{R}^{B\times C_{\text{in}}\times H_c\times W_c}$ to $y \in \mathbb{R}^{B\times C_{\text{out}}\times H_f\times W_f}$, with $C_{\text{in}} = C_{\text{out}} = 100$.
- **Width and depth.** We use `width` $= 64$ channels and stack 3 spectral blocks.
- **Lifting layer.** A 1×1 convolution maps 100 input channels to 64 channels before the spectral blocks.

- **Spectral blocks.** Each block applies a `SpectralConv2d` operator with input and output width 64 and Fourier modes $(m_1, m_2) = (16, 16)$, in parallel with a 1×1 convolution in physical space. The two outputs are summed and passed through a GELU activation. The channel dimension remains 64 across all blocks.
- **Upsampling and projection.** After the spectral blocks, we apply bilinear interpolation from (H_c, W_c) to (H_f, W_f) with scale factor s, followed by a 1×1 convolution mapping 64 channels to $C_{\text{out}} = 100$.
- **Nonlinearity and normalization.** GELU activations are used after each spectral block; no normalization layers are applied.

FNO-AR specification

- **Input and output.** The autoregressive FNO operates on the fine grid. Given a temporal context of K past fine snapshots stacked as channels, $x \in \mathbb{R}^{B\times K\times H_f\times W_f}$, it outputs K channels of the same spatial shape. In our experiments we take $K = 100$; the predicted next snapshot is read from the output channels in the usual autoregressive manner.
- **Width and depth.** We use `width` $= 64$ channels and 4 spectral blocks.
- **Lifting layer.** At each spatial location, a per-pixel linear map $\mathbb{R}^K \to \mathbb{R}^{64}$ (implemented as a 1×1 convolution) lifts the K input channels to 64 channels.
- **Spectral blocks.** Each block applies a `SpectralConv2dAR` operator with 64 input and output channels and retained modes $(m_1, m_2) = (16, 16)$, in parallel with a 1×1 convolution. The outputs are summed and, in the first three blocks, passed through a GELU activation. The channel dimension remains 64.
- **Head.** A per-pixel multi-layer perceptron (implemented as two linear layers with a GELU activation) maps $\mathbb{R}^{64} \to \mathbb{R}^K$, producing the K output channels at each spatial location.
- **Nonlinearity and normalization.** GELU activations are used in the spectral blocks and in the head MLP; no normalization layers are applied.

Model training. We selected all configurations via standard hyperparameter sweeps and report, for each method, the best validation configuration. All models used Adam [86] with cosine–annealed learning rates; the default peak rate was $\eta \in \{1\times10^{-3}, 5\times10^{-4}, 3\times10^{-4}\}$ and weight decay $\in \{0, 10^{-4}\}$. Phase 1 was trained for

5000 epochs and Phase 2 for 2500 epochs and the checkpoint with the lowest validation error was kept for Phase 1 and fine-tuned for Phase 2. For convolutional SR baselines (U-Net, EDSR) we swept depth (encoder/decoder blocks) $\in \{2, 4, 6\}$, base channels $\in \{32, 64, 128\}$, residual blocks per stage $\in \{4, 8, 16\}$, kernel size $\in \{3, 5\}$, upsampler (pixel-shuffle vs nearest neighbor), normalization (none vs. batch), activation (ReLU vs. GELU), and dropout $\in \{0, 0.05, 0.1\}$. For Fourier models (FUnet, FNO-SR, FNO-AR) we swept layer count $\in \{4, 6, 8\}$, hidden width $\in \{32, 64, 128\}$, Fourier truncation per axis $k_x, k_y \in \{8, 16, 32\}$, and activation (ReLU vs. GELU). Once a set of parameters is chosen, training SRaFTE models in Phase 1 and Phase 2 with all models tested took approximately 3 GPU hours (5000 + 2500 epochs on 1000 samples) on a NERSC Perlmutter GPU node with 1 80GB Nvidia A100.

A.4 Dataset generation

We generate paired fine–coarse datasets for three canonical PDEs on periodic domains. Both resolutions are advanced with the same time step Δt.

Navier–Stokes (2D vorticity). For each trajectory, the fine–grid initial vorticity ω_0 is drawn i.i.d. from $\mathcal{N}(0, 1)$, transformed to Fourier space, sharply low–pass filtered to modes $\{|k_x|, |k_y| \leq 16\}$, real-part taken, and inverse–transformed to yield a band–limited field. The dynamics

$$\partial_t \omega = -u \cdot \nabla \omega + \nu \Delta \omega + f, \qquad u = \nabla^\perp \psi, -\Delta \psi = \omega,$$

are advanced with a pseudo–spectral forward–Euler scheme under periodic boundary conditions. Parameters: viscosity $\nu = 10^{-4}$, step size $\Delta t = 0.01$, and forcing $f(x, y) = 0.025\Big[\sin\Big(2\pi(x + y)\Big) + \cos\Big(2\pi(x + y)\Big)\Big]$. The identical integrator is applied on both spatial resolutions.

Heat equation. The fine-grid initial temperature $u_0(x, y) \sim \mathrm{U}(0, 1)$; the coarse initial field is obtained by point decimation of u_0. Each Fourier mode (k, ℓ) is updated with the exact integrating–factor step using diffusivity $\nu = 10^{-3}$, forcing amplitude $\alpha = 10^{-2}$, and $\Delta t = 0.01$. The update is diagonal in Fourier space, so the same routine is used at both resolutions without additional dealiasing.

Wave equation. The initial velocity is $g_0 \equiv 0$ and the initial displacement is

$$f_0(x, y) = \sum_{m=1}^{8} A_m \sin\Big(2\pi(k_{x,m} x + k_{y,m} y) + \varphi_m\Big),$$

with amplitudes $A_m \sim \mathrm{U}[0.5, 1.0]$, wavenumbers $(k_{x,m}, k_{y,m}) \in \{1,2,3\}^2$, and phases $\varphi_m \sim \mathrm{U}[0, 2\pi)$. Each trajectory is linearly rescaled to $[0,1]$. Time integration uses an explicit leap–frog scheme with the finite–difference Laplacian Δ_h and periodic boundaries; parameters are wave speed $c = 0.5$ and $\Delta t = 0.01$, yielding $\mathrm{CFL} = c\,\Delta t\sqrt{2} < 1$.

A.5 Metrics

To assess the performance of all models and baselines, we adopt the following metrics:

Relative L_2 error. For a predicted $\tilde{u}$ and the corresponding true u on the same $N_f \times N_f$ grid, the *relative L_2 error* is

$$\mathcal{E}(t) = \frac{\| \tilde{u}(\cdot,\cdot,t) - u(\cdot,\cdot,t) \|_2}{\| u(\cdot,\cdot,t) \|_2}, \tag{A.4}$$

where the Euclidean norm is taken over all spatial grid points. In the following results, unless otherwise indicated, we report the mean over every time snapshot t_m.

Structural similarity index measure (SSIM). The single-scale SSIM between a prediction and the reference is

$$\mathrm{SSIM}(t) = \frac{(2\mu_u\mu_{\tilde{u}} + C_1)\,(2\sigma_{u\tilde{u}} + C_2)}{(\mu_u^2 + \mu_{\tilde{u}}^2 + C_1)\,(\sigma_u^2 + \sigma_{\tilde{u}}^2 + C_2)},$$

where $\mu_u = \mathbb{E}[u]$ and $\mu_{\tilde{u}} = \mathbb{E}[\tilde{u}]$ are local pixel-sample means, $\sigma_u^2 = \mathbb{E}[(u-\mu_u)^2]$ and $\sigma_{\tilde{u}}^2 = \mathbb{E}[(\tilde{u}-\mu_{\tilde{u}})^2]$ are local pixel-sample variances, $\sigma_{u\tilde{u}} = \mathbb{E}[(u-\mu_u)(\tilde{u}-\mu_{\tilde{u}})]$ is the local pixel-sample covariance, and $C_1, C_2 > 0$ are small constants to stabilize division.

Spectral mean-squared error (Spectral MSE). Denote the 2D Fourier amplitudes by $\widehat{u}_{k,\ell} = \mathrm{DFT}[u]_{k,\ell}, \widehat{\tilde{u}}_{k,\ell} = \mathrm{DFT}[\tilde{u}]_{k,\ell}$. The spectral MSE is $\mathrm{SpecMSE}(t) = \frac{1}{N_f^2}\sum_{k,\ell}\Big|\,|\widehat{\tilde{u}}_{k,\ell}| - |\widehat{u}_{k,\ell}|\Big|^2$. Physically, this is the mean-squared difference between spectral magnitudes. It penalizes mismatches in the distribution of energy over wavenumbers.

Pearson linear correlation r. If we flatten each field into a vector, the Pearson coefficient r is given as

$$r(t) = \frac{\sum_i (u_i - \bar{u})\,(\tilde{u}_i - \bar{\tilde{u}})}{\sqrt{\sum_i (u_i - \bar{u})^2}\sqrt{\sum_i (\tilde{u}_i - \bar{\tilde{u}})^2}},$$

where bars denote sample means. Higher values indicate higher linear correlation.

A.6 Details for 1D1V VP data generation

For the 1D1V VP time integrator, we use a second-order Lax-Wendroff (LW) scheme. Compared to Runge-Kutta 4 (RK4), we find that the LW scheme better preserves the phase-space structure in the long-time limit. For a scalar advection equation, $\partial_t f + a\,\partial_x f = 0$, the LW update is given by,

$$f_i^{n+1} = f_i^n - \frac{a\,\Delta t}{2\Delta x}\Big(f_{i+1}^n - f_{i-1}^n\Big) + \frac{a^2\Delta t^2}{2\Delta x^2}\Big(f_{i+1}^n - 2f_i^n + f_{i-1}^n\Big).$$

In this work, we will consider a (2 + 1)-dimensional system with one real spatial component and one velocity spatial component (1D1V). We study two types of systems that arise as a result of changing the initial distribution of electrons: (i) randomized electron distribution and (ii) the two-stream instability. For the randomized electron distribution, we fix the ions (see Appendix A.7) with a Maxwell-Boltzmann distribution given by,

$$f_i(x,v,0) = \frac{1}{\sqrt{2\pi\, v_{\text{th,i}}^2}}\exp\Big(-\frac{v^2}{2\,v_{\text{th,i}}^2}\Big),$$

and initialize the electrons with a randomized cosine-superposition given by,

$$f_e(x,v,0) = \Big[1 + \sum_{n_x,n_v=0}^{m_x-1,m_v-1} \cos\Big(2\pi(n_x\,x + n_v\,v) + \varphi_{n_x,n_v}\Big)\Big]^2.$$

Here, $v_{\text{th,i}}$ represents the thermal velocity (average speed of ions moving randomly within the plasma due to their temperature), the phases φ_{n_x,n_v} are drawn i.i.d. in $[0,2\pi)$, and the result is renormalized to unit density. In practice, we fix $m_x, m_v = \{(4,4),(8,8)\}$, and generate 1000 trajectories each by randomly sampling φ_{n_x,n_v}. We refer to the dataset with $m_x, m_v = 4,4$ as the low frequency dataset, and the dataset with $m_x, m_v = 8,8$ as the high frequency dataset. We do not adaptively timestep, as this poses some slight complications for Phase 2 training and future time extrapolation. We instead fix $\Delta t = 0.01$, which satisfies the CFL condition, and propagate until physical time $T_{train} = 1.0$. For the subsequent results, we assume a 32×32 grid for the coarse grain data generation and a 128×128 grid for the fine grain data generation.

For the two-stream dataset, the ions are taken to be stationary Maxwell-Boltzmann distributions in the same way used for the randomized dataset. The electron distribution is a pair of counter-propagating beams whose density is perturbed by a small, broadband

perturbation:

$$f_e(x,v,0) = \Big[1 + A_1\cos(kx+\phi_1) + A_2\cos(2kx+\phi_2)\Big]\,\frac{1}{2\sqrt{2\pi}} \cdot \left[e^{-\frac{(v-\bar{v})^2}{2}} + e^{-\frac{(v+\bar{v})^2}{2}}\right].$$

Here, $k = \beta/\lambda_D$ with λ_D the Debye length, $A_1 = \varepsilon$ is the fundamental-mode amplitude, $A_2 = \varepsilon r$ is the second-harmonic amplitude, and $\bar{v}$ is the beam drift speed. Each trajectory is obtained by independently sampling the parameter set

$$\begin{aligned} \varepsilon = \varepsilon_0(1+\delta_\varepsilon), \quad \delta_\varepsilon \sim \mathrm{U}[-0.10, 0.10], \quad \varepsilon_0 = 5\times 10^{-3}, \\ \beta = \beta_0 + \delta_\beta, \quad \delta_\beta \sim \mathrm{U}[-0.05, 0.05], \quad \beta_0 = 0.20, \\ \bar{v} = \bar{v}_0(1+\delta_v), \quad \delta_v \sim \mathrm{U}[-0.05, 0.05], \quad \bar{v}_0 = 2.4, \\ r \sim \mathrm{U}[0.05, 0.10], \qquad \phi_1, \phi_2 \sim \mathrm{U}[0, 2\pi). \end{aligned}$$

This ensemble therefore varies the perturbation strength, the effective wavenumber, the beam drift, the second-harmonic content, and the phases. This creates a dataset that captures a wide range of growth and saturation behaviors. We generate 1000 trajectories for training and evolved each one on a coarse 64×64 grid and a fine 256×256 grid using a fixed timestep $\Delta t = 0.01$ and simulate until physical time $T_{\text{sim}} = 50$. During post-processing, we retain only the one-second slice $22 \le t < 23$, for both grids, resulting in 101 equally spaced frames. This captures the portion in time evolution when the electron two-stream instability has grown well enough into its nonlinear regime, but has yet to fully mix/form coherent vortices.

Finally, we note that both datasets used uniform grid spacing where $x \in [-2\pi, 2\pi]$ and $v \in [-6, 6]$.

A.7 Why freezing the ions is valid

A *stationary-ion* (single-species) Vlasov–Poisson model is routinely adopted whenever the dynamics of interest unfold on electronic time-scales. Canonical examples of interest from the plasma community include linear and nonlinear Landau damping [9]. Landau damping is the collisionless attenuation of electrostatic waves due to phase-mixing in the *electron* distribution function. This process unfolds on electronic time scales and therefore treats the far heavier ions as effectively immobile. More recently, data-driven studies of Vlasov–Poisson systems using latent-space identification

or PINNs [64, 139] have also employed this approximation to make predictions with respect to the electron dynamics.

The rationale is the large separation between the electron and ion plasma periods, quantified by the large discrepancies between their masses as

$$\frac{T_i}{T_e} = \sqrt{\frac{m_i}{m_e}},$$

where m_e and m_i are the electron and ion masses and T_e and T_i their respective plasma periods. In general, the plasma frequency of a species with number density n_s, charge magnitude q_s (for electrons $q_e = -1$), and mass m_s is

$$\omega_{p,s} = \sqrt{\frac{n_s q_s^2}{\epsilon_0 m_s}},$$

so the period $T_s = 2\pi/\omega_{p,s}$ scales as $T_s \propto \sqrt{m_s}$; hence $T_i/T_e = \sqrt{m_i/m_e}$, showing that heavier ions oscillate intrinsically more slowly than light electrons under the same electrostatic restoring force [55]. Even with a reduced mass ratio $m_i/m_e = 25$ one obtains $T_i = 5\,T_e$. The electron period is $T_e = 2\pi/\omega_{pe}$ with

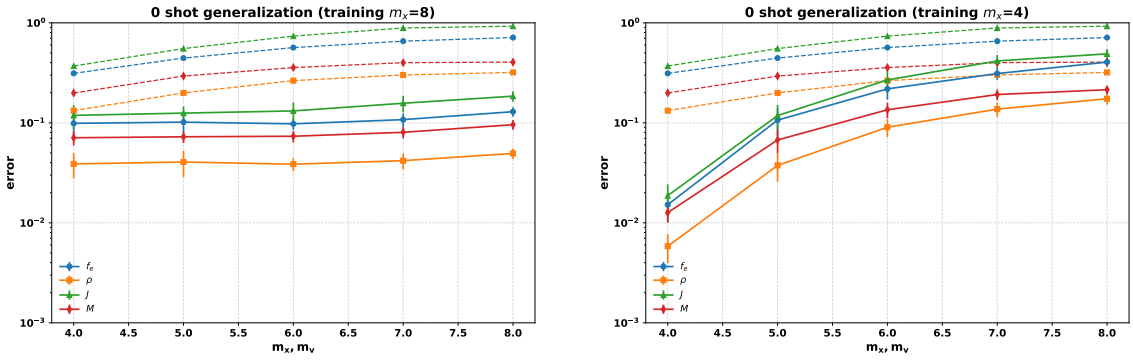
$$\omega_{pe} = \sqrt{\frac{n_e e^2}{\epsilon_0 m_e}},$$

where n_e is the electron number density, e the elementary charge, and ϵ_0 the vacuum permittivity (in our normalized units $n_e = 1$ and $e = \epsilon_0 = 1$, giving $\omega_{pe} = 1$). Over simulation windows of only a few electron periods, the majority of the ion velocity changes negligibly, so omitting the ion species from Vlasov–Poisson equation leaves the linear growth, nonlinear saturation, and phase-space structure of the electron species virtually unchanged. A full two-species treatment becomes essential only when simulations extend well beyond T_i or deliberately excite coupled electron-ion modes such as the Buneman instability[55, 9]; such regimes lie outside the methodological scope of this work.

A.8 Supplementary results

A.8.1 Training frequency transfer

The left panel of Figure A.2 shows a sweep of test set errors where $m_x = m_v \in \{4, \ldots, 8\}$ using a model trained *only* on the highest band-limit $(8, 8)$. The



(a) Training on the *high-frequency* dataset $(m_x, m_v) = (8, 8)$ and evaluating on progressively lower test bandwidths.

(b) Training on the *low-frequency* dataset $(m_x, m_v) = (4, 4)$ and evaluating on progressively higher test bandwidths.

Figure A.2: Zero-shot generalization sweeps for the 1D1V Vlasov–Poisson super-resolution task with random cosine superposition initial condition. Solid curves show the FUnet prediction errors while dashed curves indicate the bicubic upsampling baseline. High-frequency training (left) maintains low error when asked to resolve coarser spectra, whereas the converse (right) incurs rapidly growing error when tested with high-frequency data.

dashed curves correspond to the bicubic interpolation baseline errors, and the solid curves represent the FUnet predictions errors. Surprisingly, the error appears to decrease as the gap between train- and test-frequencies widen. Nevertheless, we see that for all predictions of the flow and moments, that the FUnet predictions retain anywhere from a $7 - 10\times$ improvement in error over the baseline, demonstrating the non-trivial generalization to out of distribution spectra. The right panel shows a model that is trained only on the lowest band-limit $(4, 4)$. As expected, the error increases as the frequency band-width increases.

To understand the transferability, we investigate the weights of the model for insight. Figure A.3 shows that every weight matrix in the high-frequency network (trained on $(m_x, m_v) = (8, 8)$ data) has a spectral norm that matches or exceeds its low-frequency analogue in key layers that are responsible for generating fine-scale features. The gap is noticeably the largest in the global Fourier layer `bottleneck.1`, which is primarily responsible for processing the mixed Fourier modes. Thus, we can reason that the high-frequency model retains more coarse directions, whilst additionally activating higher-frequency degrees of freedom that do not appear during $(m_x, m_v) = (4, 4)$ training.

Moment	$(m_x, m_v)^{\text{train}} \to (m_x, m_v)^{\text{test}}$	Baseline	SRaFTE model
		Bicubic	FUnet
ρ	$(4,4) \to (4,4)$	1.32×10^{-1}	$\mathbf{6.42\times10^{-3}} \pm 1.52\times10^{-3}$
	$(8,8) \to (8,8)$	3.19×10^{-1}	$\mathbf{4.81\times10^{-2}} \pm 5.83\times10^{-3}$
	$(8,8) \to (4,4)$	1.32×10^{-1}	$\mathbf{3.58\times10^{-2}} \pm 1.01\times10^{-2}$
J	$(4,4) \to (4,4)$	3.69×10^{-1}	$\mathbf{1.87\times10^{-2}} \pm 4.89\times10^{-3}$
	$(8,8) \to (8,8)$	9.25×10^{-1}	$\mathbf{1.77\times10^{-1}} \pm 1.99\times10^{-2}$
	$(8,8) \to (4,4)$	3.69×10^{-1}	$\mathbf{1.08\times10^{-1}} \pm 3.00\times10^{-2}$
M	$(4,4) \to (4,4)$	1.98×10^{-1}	$\mathbf{1.28\times10^{-2}} \pm 3.53\times10^{-3}$
	$(8,8) \to (8,8)$	4.04×10^{-1}	$\mathbf{9.17\times10^{-2}} \pm 9.52\times10^{-2}$
	$(8,8) \to (4,4)$	1.98×10^{-1}	$\mathbf{6.49\times10^{-2}} \pm 1.08\times10^{-2}$

Table A.2: **Randomized 1D1V Vlasov Phase 1 test set relative L_2 errors for frequency transfer.** Errors are averaged over 20 new initial conditions. FUnet results are averaged over five training runs and reported as mean$\pm$standard deviation, while bicubic upsampling is deterministic and reported as a mean only. Lower values are highlighted in bold.

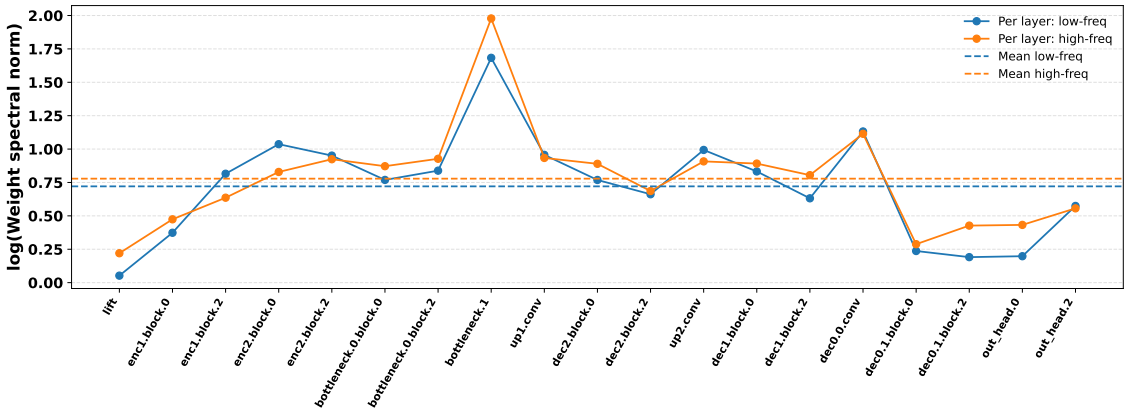


Figure A.3: **Per-layer spectral norms.** Each data point is the largest singular value $\|W\|_2$ of a weight matrix in the model. Larger values therefore signal a greater ability to pass or create high-frequency features. Both models exhibit to clear spikes: one at the Fourier mixing layer `bottleneck.1` and one at the first decoder convolution `dec0.0.conv`. These are precisely the layers that must inject or preserve fine-scale modes as the feature map is downsampled and then re-expanded.

Let f_θ be a trained network with parameters θ and select a single weight matrix $W_\ell \subset \theta$.

Given N coarse snapshots $\{x_i\}_{i=1}^N$ we form for each i,

$$g_i = \frac{\partial}{\partial W_\ell}\Big[\mathcal{L}\big(f_\theta(x_i)\big)\Big] \in \mathbb{R}^P, \qquad P = |W_\ell|,$$

for $\mathcal{L}$ the loss. If we then stack the parameter gradients row-wise, we obtain the Jacobian, $J = [g_1^\top \dot{s} g_N^\top]^\top \in \mathbb{R}^{N\times P}$, and the *layer-restricted neural tangent kernel (NTK)* [69] is

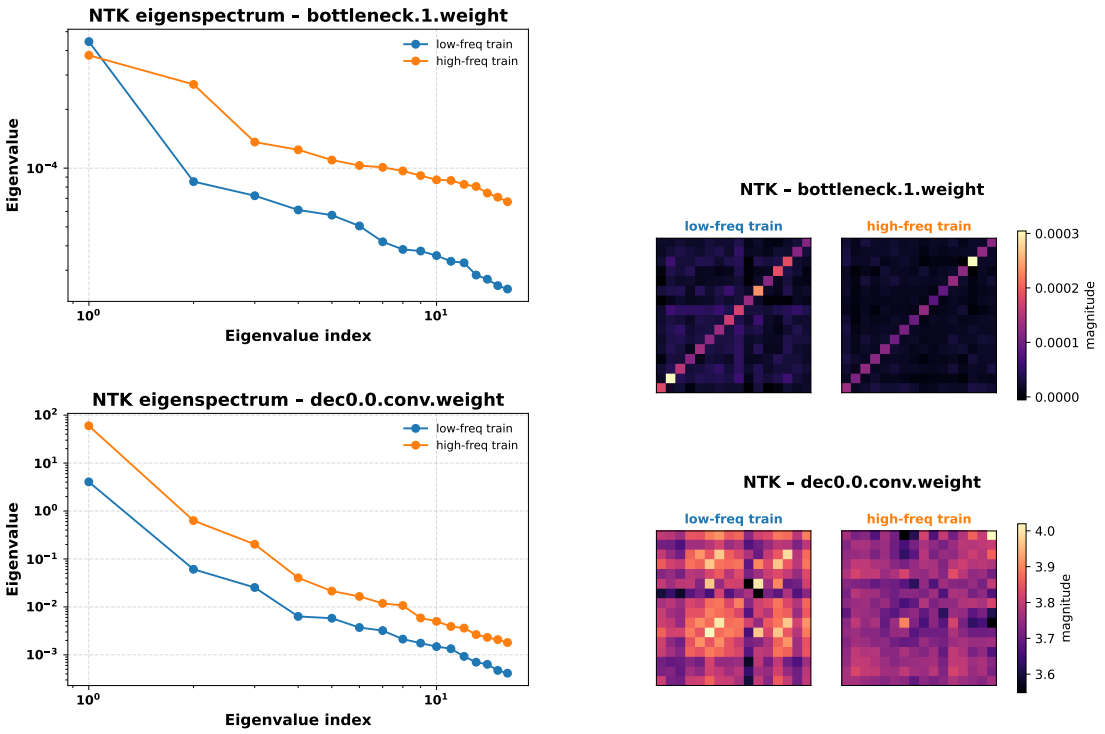
$$K = J\,J^\top \in \mathbb{R}^{N\times N}, \quad K_{ij} = \langle g_i, g_j\rangle.$$

Hence K measures how similar the weight update would be if the network were (re-)trained on snapshots x_i and x_j. We will make use of this to analyze two NTK diagnostics [175]:

1. **Eigenspectrum.** The sorted eigenvalues $\lambda_1 \geq \cdots \geq \lambda_N$ of K help quantify the effective dimensionality of the subspace generated by the span of the gradients: a rapid decay will indicate that only a few directions in the parameter space are being adjusted during training, whereas a flatter tail indicates a richer set of learnable directions.

2. **Kernel heatmap.** Visualizing K itself highlights whether gradients are independent or redundant (i.e. K is diagonal dominated or has strong off-diagonal blocks). A nearly diagonal kernel implies that different coarse snapshots possess distinct parameter updates, thus are likely to be well represented without the need for fine-tuning.

The NTK diagnostics in Figure A.4 corroborate the viewpoint that high-frequency training can generalize to low-frequency testing, but not vice-versa. For the bottleneck Fourier layer and the first decoder convolution, the eigenspectrum of the high-frequency training remains order(s) of magnitude larger beyond the first few ranks and its kernel is nearly diagonal. This indicates that for distinct snapshots, there are linearly independent gradients. Conversely, the low-frequency training eigenspectra collapse after rank $k \approx 4$, and the kernels have pronounced off-diagonal blocks. This signals that many input directions share identical or near identical curvature. These kernel-level differences explain the performance sweep in Figure A.2: a new low-frequency snapshot will likely lie inside the curvature subspace spanned by the high-frequency NTK and is therefore recovered immediately in evaluation, whereas a high-frequency snapshot falls outside of the low-frequency span, so the model fails to generalize.



(a) NTK eigenspectra (b) NTK heatmaps

Figure A.4: **NTK statistics** *Top row*: Fourier mixing layer `bottleneck.1`; *bottom row*: first decoder convolution `dec0.0.conv`. High-frequency training (orange) yields richer eigenspectra and nearly diagonal kernels, whereas low-frequency training (blue) collapses to a low-rank span with shared gradients.

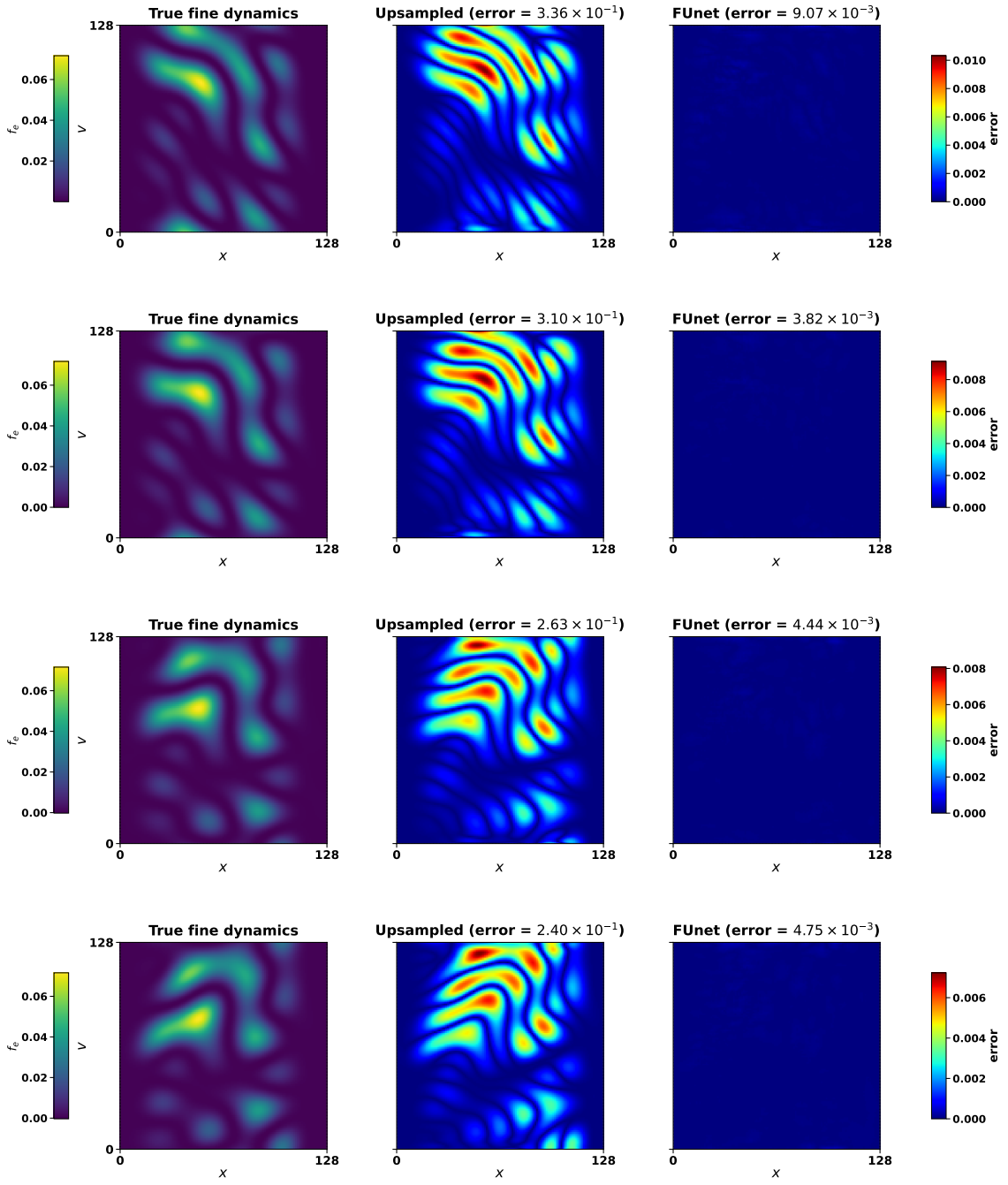


Figure A.5: Total Phase 1 super-resolution for randomized dataset.

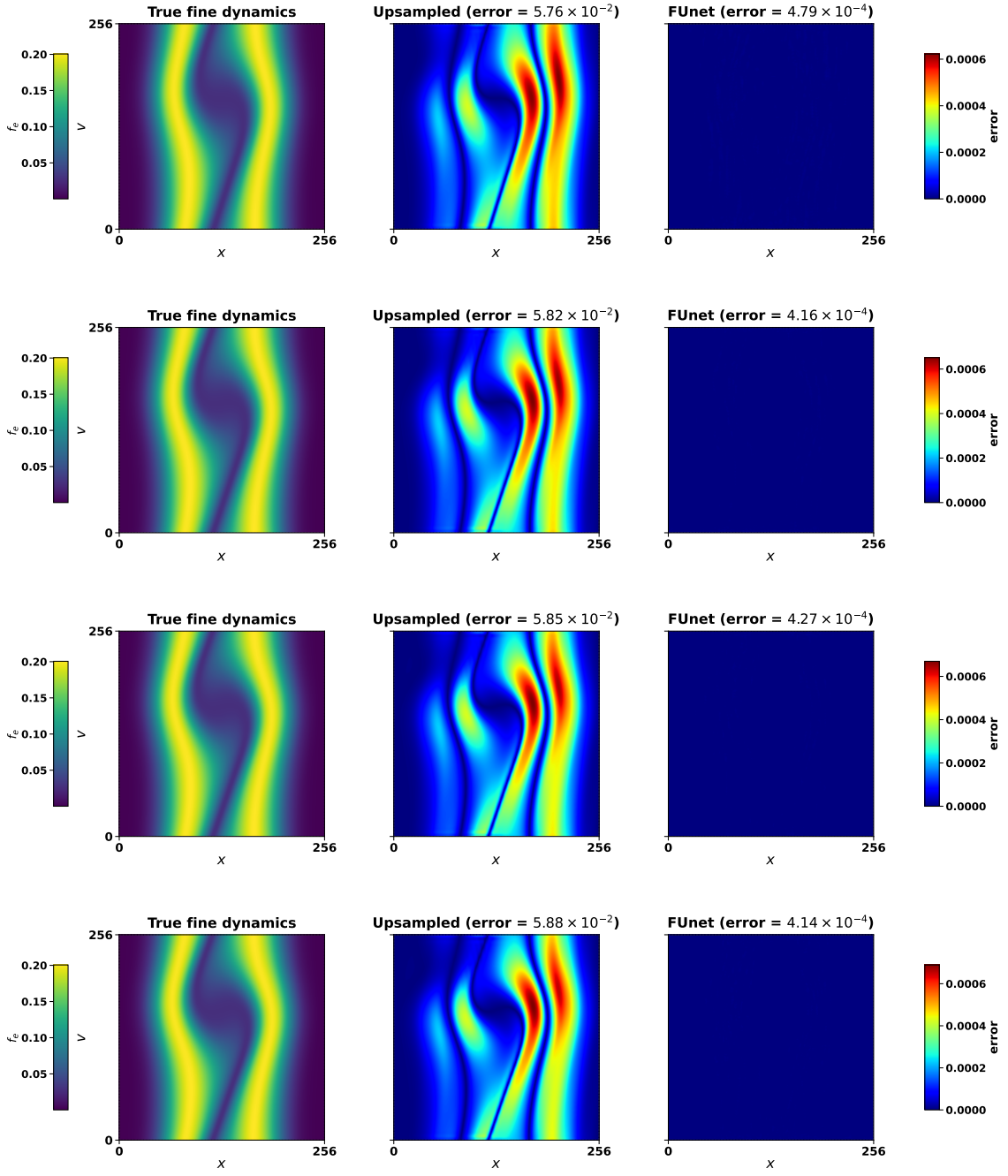


Figure A.6: Total Phase 1 super-resolution for two-stream instability.

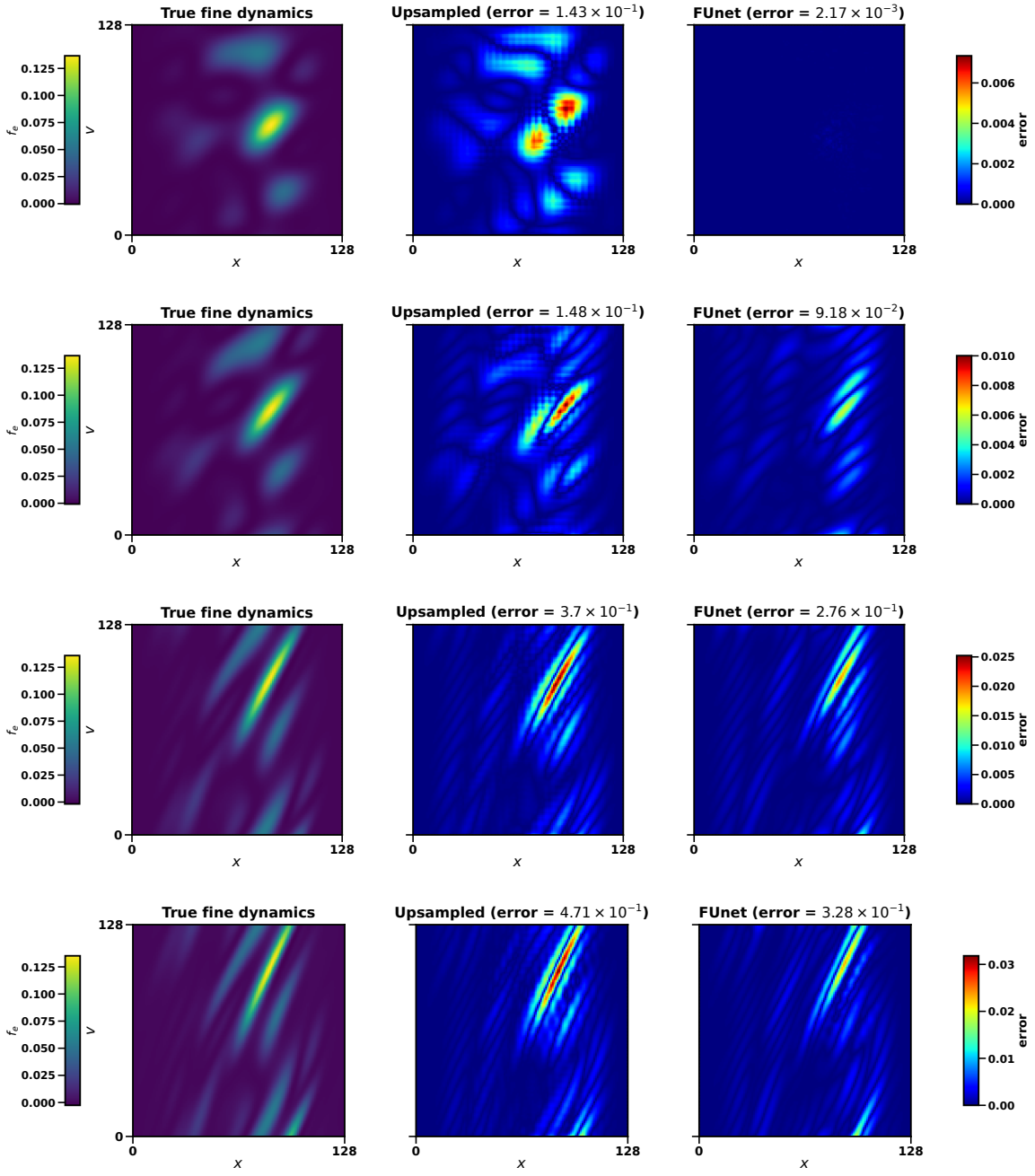


Figure A.7: Total Phase 2 extrapolation for randomized dataset

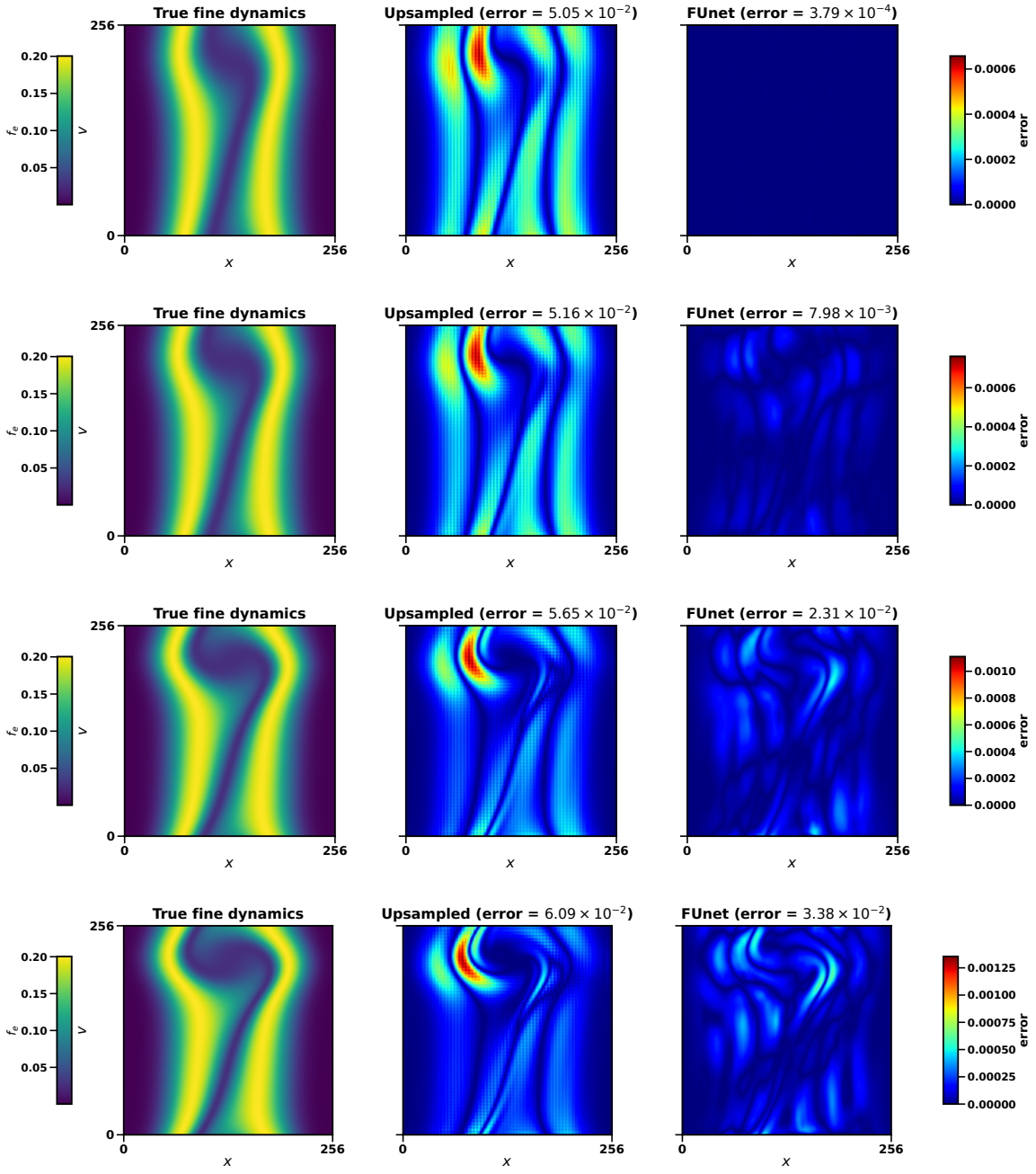


Figure A.8: Total Phase 2 extrapolation for two-stream instability.

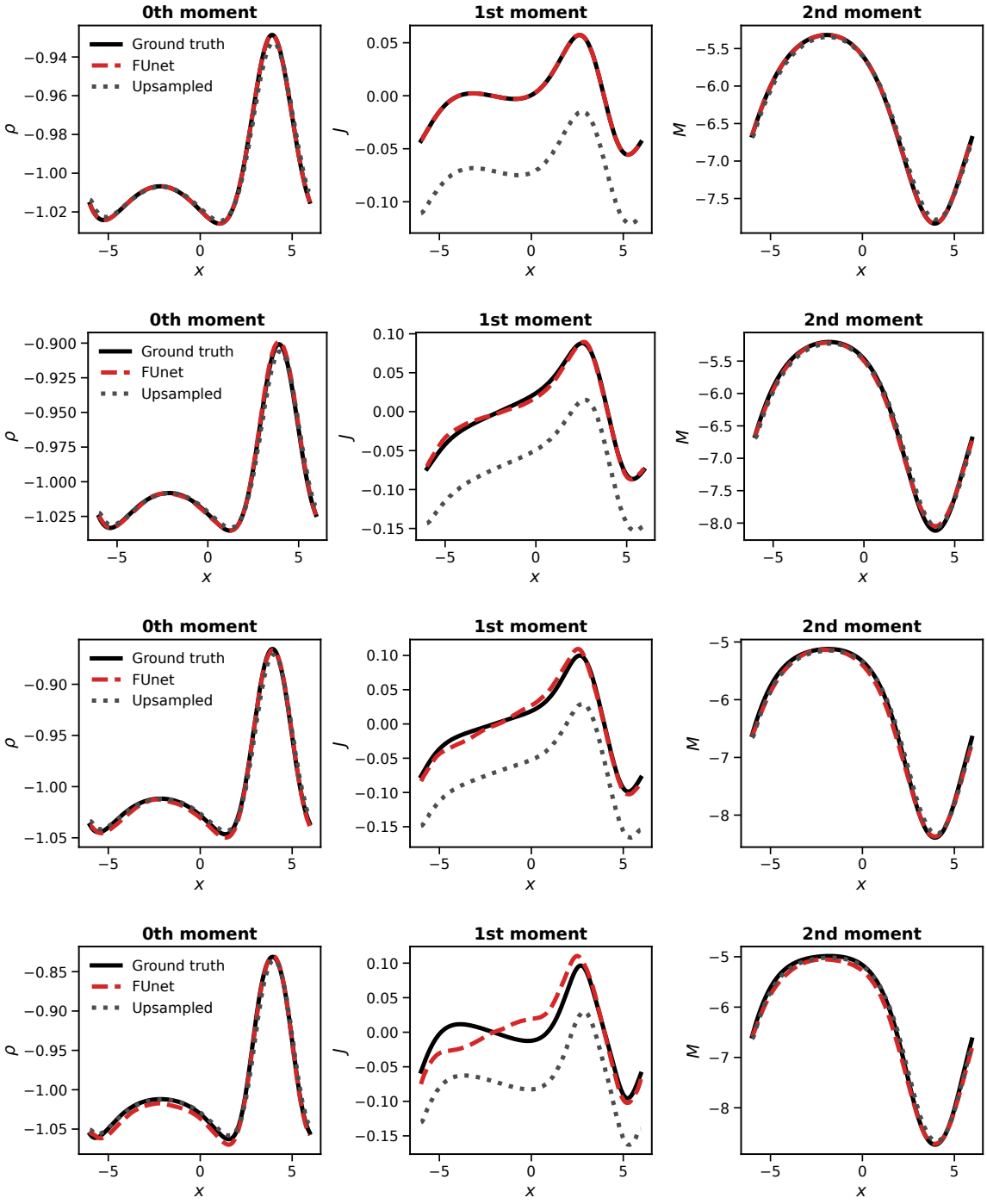


Figure A.9: Total Phase 2 moments extrapolation for two-stream instability.

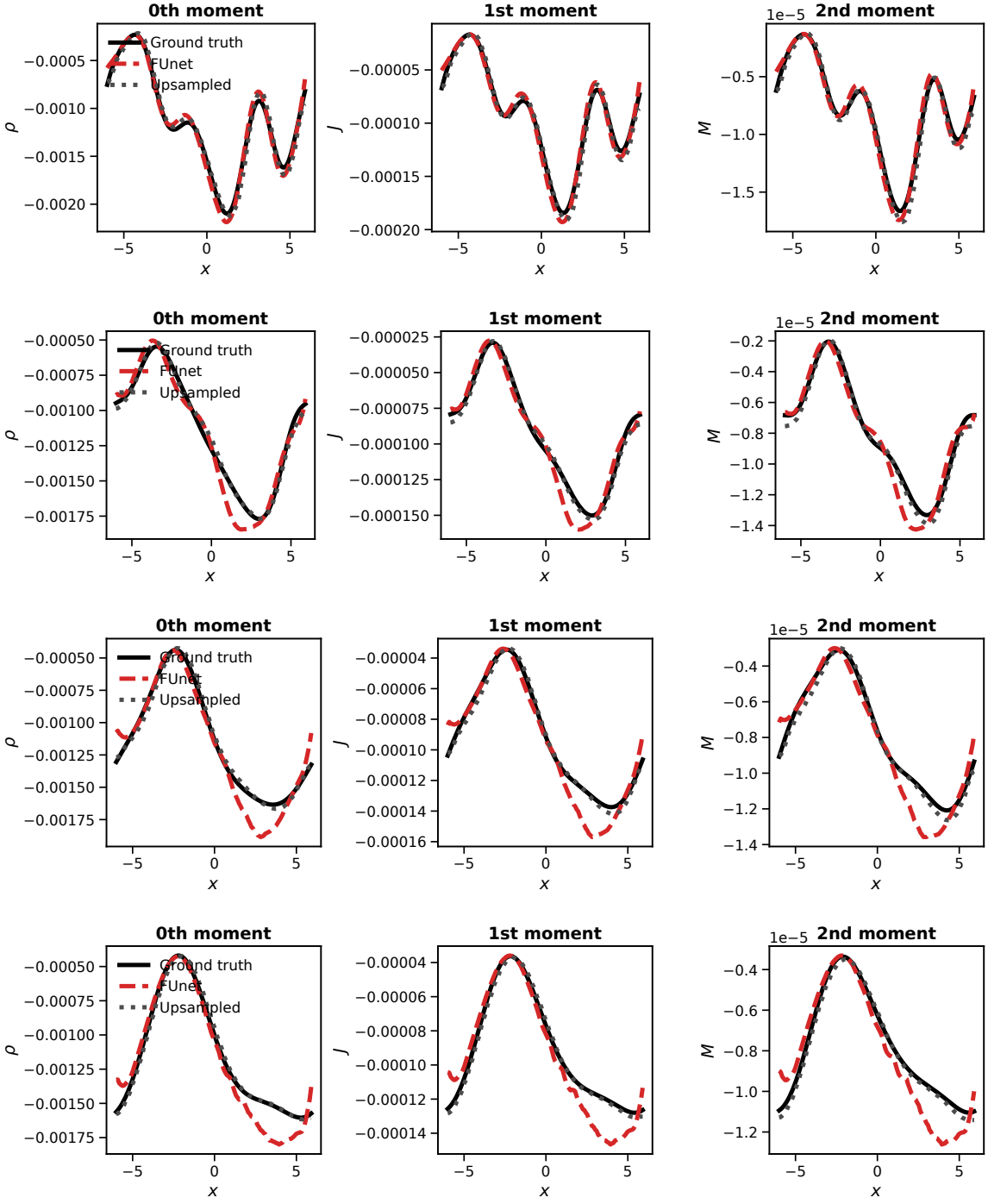


Figure A.10: Total Phase 2 moments extrapolation for randomized dataset

Appendix B

Appendix for Chapter 4

B.1 A case study of quantum resource estimates

Here, we estimate the quantum resource requirements of FDODMD/ODMD on realistic quantum hardware. For simplicity, let us consider the example of LiH in the STO-3G basis set. For this case, Li fixed at the origin and H placed 1.595 Å along the z-axis.

Within this basis set, we find that Li contributes 5 spatial molecular orbitals (MOs) and H contributes 1. If we freeze the core electron of Li, this leaves 5 active spatial MOs, which implies the number of qubits needed to represent the effective Hamiltonian is **10**.

The second-quantized electronic Hamiltonian is

$$H = \sum_{p,q} h_{pq}\, a_p^\dagger a_q + \frac{1}{2} \sum_{p,q,r,s} h_{pqrs}\, a_p^\dagger a_q^\dagger a_r a_s.$$

Here p, q, r, s index an orthonormal spin–orbital basis $\{\phi_p\}$; $a_p^\dagger$ (a_p) creates (annihilates) an electron in ϕ_p and obeys $\{a_p, a_q^\dagger\} = \delta_{pq}$. The coefficients are $h_{pq} = \langle \phi_p | -\frac{1}{2}\nabla^2 - \sum_A Z_A/|\mathbf{r} - \mathbf{R}_A| | \phi_q \rangle$ (one-electron integrals) and $h_{pqrs} = \langle pq \| rs \rangle$ (two-electron Coulomb integrals). Under the standard Jordan–Wigner fermion-to-qubit mapping and using Qiskit [70] for circuit compilation, the effective Hamiltonian contains **631 unique Pauli strings**.

In the benchmarks that we consider, we manually prescribe the spectral weights $\{p_n\}$ used in forming (4.3). We fix a target ground-state probability p_0 and distribute the remainder uniformly over excitations. This controls the conditioning of the identification

problem and probes K-convergence behavior under the spectrum. This is a simulation artifact, not a hardware prescription: realizing the specific $\{p_n\}$ exactly would indeed require prior eigenbasis knowledge. In general, preparing a sparse linear combination of computational basis states incurs a gate complexity that scales linearly with the sparsity and polynomially with the system size. On hardware, a practical computation would use a Hartree-Fock reference state (**0 CNOTs**) to obtain the reference state to be used for time evolution for ODMD/FDODMD. Notably, because ODMD/FDODMD are not dependent on the exact overlap probabilities themselves — only requiring non-zero overlap with the frequencies of interest — our resource estimates here are for practical state preparations.

On hardware, we require a compiled single step $U_{\Delta t} \approx e^{-iH\Delta t}$, applied repeatedly. In principle, we can compile $U_{\Delta t}$ with a first-order product formula using M Trotter steps, which forms an intrinsic microstep $\Delta t/M$:

$$U_{\Delta t} \approx [S_1(\Delta t/M)]^M, \quad S_1(\tau) = \prod_{\nu=1}^{\Gamma} e^{-ic_\nu P_\nu \tau}.$$

Compilation cost per step is then $O(M\Gamma)$. A CNOT upper bound per exponential is then given by $2(\mathrm{w}(P_\nu) - 1)$, where $\mathrm{w}(\cdot)$ is a Pauli weight. This follows from the standard parity-ladder construction [202]: conjugate non-Z Paulis to Z, compute the parity of the $w(P_\nu)$ participating qubits onto a pivot with $w(P_\nu) - 1$ CNOTs, apply a single rotation R_z on the pivot, and uncompute with another $w(P_\nu) - 1$ CNOTs (ignoring one-qubit gates). Thus, we have that

$$\text{CNOTs per } U_{\Delta t} \lesssim M \sum_{\nu=1}^{\Gamma} [2(\mathrm{w}(P_\nu) - 1) + 2].$$

Since we use the Hadamard test, evolution is controlled by an additional ancilla qubit, thus there are 2 additional CNOTs per term. If we take $M = 1$, this yields $\mathbf{4\,956}$ **(uncontrolled)** or $\mathbf{6\,216}$ **(controlled) CNOTs** per $U_{\Delta t}$.

With temporal samples at $t_k = k\Delta t$ for $1 \le k \le K$, the deepest circuit occurs at $k = K$ and is composed of $K\times$(CNOTs per $U_{\Delta t}$) two-qubit gates. The maximum number of CNOTs then at $k = K = 1500$ is then roughly $\mathbf{7.434 \times 10^6}$ **uncontrolled** or $\mathbf{9.324 \times 10^6}$ **controlled**. Furthermore, the total number of measurement shots for collecting the observable time series $\mathrm{Re}\, s(t_k)$ scales as $(K + 1)/\varepsilon^2$. Taking $K = 1500, \epsilon = 0.1$ means roughly $\mathbf{1.5 \times 10^5}$ **shots** are needed.

The estimates here assume for the hardware all-to-all connectivity, no cancellation, and no error mitigation techniques; transpilation/optimizations can only reduce the estimates. Fig. B.1 visualizes the computational load for this particular unoptimized case study and it is shown that the CNOT cost is dominated by the mid-weight Pauli strings. Proper

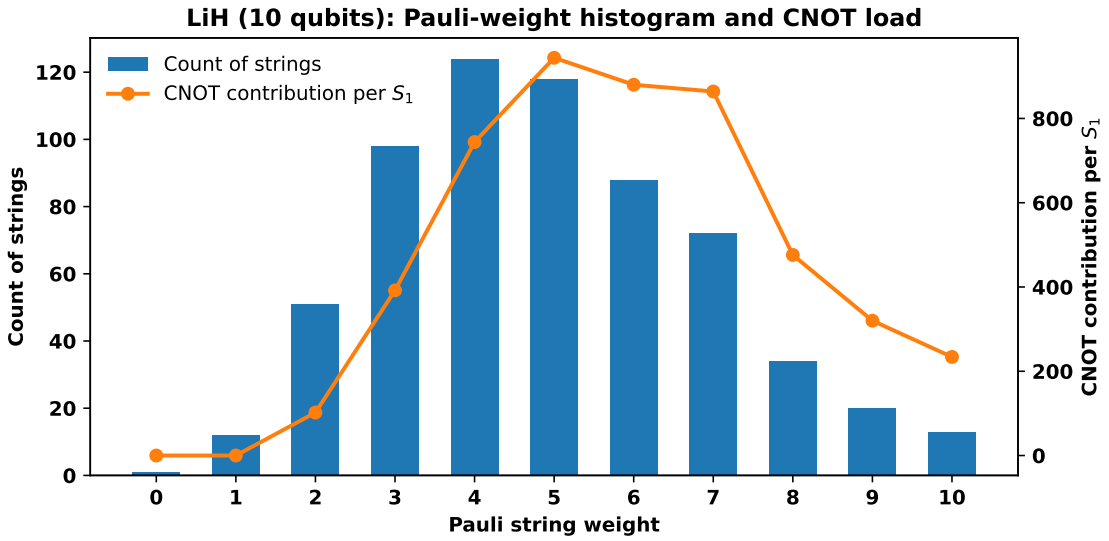


Figure B.1: LiH (STO-3G, 10 qubits): Pauli-weight histogram and CNOT load. Bars show the count of Pauli strings by weight for the Jordan-Wigner Hamiltonian; the line overlays each weight bin's contribution to the first-order microstep S_1 CNOT cost.

optimizations, such as grouping commuting Pauli strings and tapering to a fixed particle-number sector, can substantially reduce the circuit depth and overall gate overhead. In conjunction with a smaller K or choice of Δt that is consistent with the target spectral resolution, the case of simulating LiH on NISQ-era hardware is challenging but plausible. Indeed, recent studies have shown success in simulating LiH in the STO-3G and larger basis sets for VQE and excited-state variants on IBM superconducting processors and with Qiskit hardware backends/simulators [207, 157, 127, 77].

Molecules with large active spaces (e.g., Cr_2) typically admit significantly larger Hamiltonians and heavier Pauli weights, implying an increase in quantum resources. As capabilities scale, these systems will emerge as core benchmarks; addressing strongly correlated molecular targets will rank among the most exciting, high-impact tasks in the near term and early fault-tolerant era.

B.2 Optimal shot allocation

Suppose we ask: how should we allocate shots at each time t_k, from a fixed total budget of N_{total} shots, so as to minimize the uncertainty in each noisy $\mathrm{Re}\,d(t_k)$ defined in (4.4)? A general answer to this question that assumes no prior knowledge of $\mathrm{Re}\,s(t_k)$ is beyond

the scope of this chapter. However, if we take a hindsight/oracle point of view, whereby $\mathrm{Re}\, s(t_k)$ is known and we want to determine after the fact how many shots we should have used to minimize uncertainty, then a simple solution is possible.

Let us quantify the total uncertainty by the total variance of $\mathrm{Re}\, d(t_k)$ over all times. Summing (4.5) over all k and using $\pi_{t_k} = (1 + \mathrm{Re}\, s(t_k))/2$, we get $\sum_k \mathrm{Var}(\mathrm{Re}\, d(t_k)) = \sum_k (1 - \mathrm{Re}\, s(t_k)^2)/n_{t_k}$. This leads us naturally to the constrained optimization:

$$\begin{aligned} &\text{minimize} \quad \sum_k (1 - \mathrm{Re}\, s(t_k)^2)/n_{t_k} \\ &\text{subject to} \quad \sum_k n_{t_k} = N_{\text{total}} \text{ and } n_{t_k} \geq 0. \end{aligned} \tag{B.1}$$

Here we minimize with respect to n_{t_k}, the number of shots to allocate at time t_k. To solve this optimization problem, we use Lagrange multipliers, i.e., we seek critical points $(n^*_{t_k}, \mu^*)$ of the Lagrangian

$$\mathcal{L}(n_{t_k}, \mu) = \sum_k (1 - \mathrm{Re}\, s(t_k)^2)/n_{t_k} + \mu\Big(\sum_k n_{t_k} - N_{\text{total}}\Big). \tag{B.2}$$

Setting $\nabla_{n_{t_k}} \mathcal{L}(n_{t_k}, \mu) = 0$ yields $n_{t_k} = \sqrt{(1 - s(t_k)^2)/\mu}$. Combining this with the constraint $\sum_k n_{t_k} = N_{\text{total}}$ enables us to solve for the optimal μ. Substituting such optimal μ back into the n_{t_k} equation yields the optimal number of shots to allocate at time t_k:

$$\left\lfloor n^*_{t_k} \right\rfloor = N_{\text{total}} \left(\frac{\sqrt{1 - s(t_k)^2}}{\sum_k \sqrt{1 - s(t_k)^2}} \right). \tag{B.3}$$

Note that (B.3) is minimized when $s(t_k) = \pm 1$, and maximized when $s(t_k) = 0$.

We view (B.3) as justifying the use of an equal number of shots at each time t_k. In Fig. B.2, we plot the optimal number of shots $n^*_{t_k}$ at time t_k, prescribed by (B.3), for various total shot budgets $N_{\text{total}} \in \{5 \times 10^3, 10^4, 10^5, 10^6\}$. Dashed black lines indicate the uniform allocation $N_{\text{total}}/K_{\max}$ with $K_{\max} = 1000$. At time $t = 0$, $s(0) = \langle \phi_0 | \phi_0 \rangle = 1$, implying zero variance — thus no shots are needed at $t = 0$.

For every $t_k > 0$, we find in Fig. B.2 that $n^*_{t_k}$, calculated with the noiseless observable $s(t_k)$ in (B.3), is either $N_{\text{total}}/K_{\max}$ or $N_{\text{total}}/K_{\max} - 1$. Empirically, using an equal number of shots $N_{\text{total}}/K_{\max}$ at each time t_k is essentially equivalent to optimally allocating the shots by minimizing the total variance; the gap in the performance between equal and optimal shot allocation is negligible.

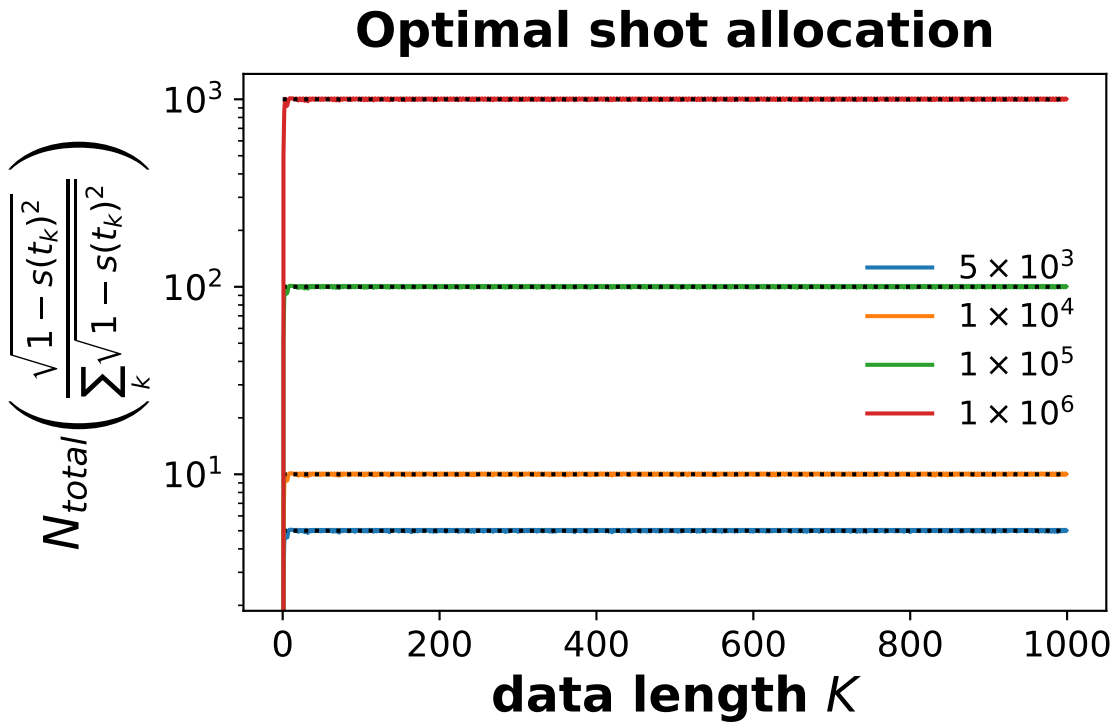


Figure B.2: Optimal shot allocation for $N_{\text{total}} = \{5 \times 10^3, 10^4, 10^5, 10^6\}$. We compute (4.2) with $(K_{\max}, \Delta t, p_0) = (1000, 1.0, 0.2)$ as input data to (B.3). We find that it is nearly optimal to allocate an equal number of shots across all time.

B.3 Hamiltonian Scaling

Let $\overline{H}$ denote the molecular Hamiltonian whose GSE must be estimated; in this chapter, this is the ASCI Hamiltonian. Let $\{\overline{E}_0, \ldots, \overline{E}_{N-1}\}$ denote the eigenvalues of $\overline{H}$; similarly, let $\{E_0, \ldots, E_{N-1}\}$ denote the eigenvalues of a linearly rescaled Hamiltonian

$$H = \beta_0 I + \beta_1 \overline{H}. \tag{B.4}$$

We assume all eigenvalues are indexed in nondecreasing order. In order to satisfy an ODMD convergence criterion [161], we seek a rescaling that implies

$$(E_{N-1} + E_1 - 2E_0)\Delta t < 2\pi. \tag{B.5}$$

Let $\underline{E}$ and $\overline{E}$ be *estimated* lower and bounds, respectively, on the true spectrum; by this we mean that for all $j = 0, 1, \ldots, N-1$, we must have

$$\underline{E} < \overline{E}_j < \overline{E}. \tag{B.6}$$

In a practical setting, $\underline{E}$ and $\overline{E}$ can be estimated as follows. First one runs a Hartree-Fock calculation on the molecular system under consideration to calculate Hartree-Fock eigenvalues $\tilde{E}_0$ and $\tilde{E}_{N-1}$ that approximate the true eigenvalues with bounds [120, 136] of the form $|\tilde{E}_0 - \overline{E}_0| < \alpha_0$ and $|\tilde{E}_{N-1} - \overline{E}_{N-1}| < \alpha_{N-1}$. Let $\alpha = \max\{\alpha_0, \alpha_{N-1}\}$ and set

$$\underline{E} = \tilde{E}_0 - \alpha \;\text{ and }\; \overline{E} = \tilde{E}_{N-1} + \alpha.$$

By construction, (B.6) will be satisfied. In the present chapter, *to simulate the above scenario*, we use the true ASCI eigenvalues with a large value of α (in particular, $\alpha = 0.2$) and set $\underline{E} = \overline{E}_0 - \alpha$ and $\overline{E} = \overline{E}_{N-1} + \alpha$. Now define $\Delta E = \overline{E} - \underline{E}$ and $\mu_E = (\overline{E} + \underline{E})/2$. We then use

$$\beta_0 = -\pi\mu_E(2\Delta t \Delta E)^{-1} \;\text{ and }\; \beta_1 = \pi(2\Delta t \Delta E)^{-1}.$$

Let us show that this rescaling satisfies (B.5). First, since $\overline{H}$ is Hermitian, there exists a unitary V such that $\overline{H} = V\overline{\Lambda}V^\dagger$ where $\overline{\Lambda}$ is diagonal and contains the eigenvalues of $\overline{H}$. Note that H is also diagonalized by V; by (B.4), $V^\dagger H V = \beta_0 I + \beta_1 \overline{\Lambda}$, which is purely diagonal. Thus the eigenvalues of H and $\overline{H}$ are related via

$$E_j = \beta_0 + \beta_1 \overline{E}_j.$$

Using this and (B.6), we have

$$E_{N-1} - E_0 = \frac{\pi}{2\Delta t}\frac{\overline{E}_{N-1} - \overline{E}_0}{\overline{E} - \underline{E}} < \frac{\pi}{2\Delta t}$$

$$E_1 - E_0 = \frac{\pi}{2\Delta t}\frac{\overline{E}_1 - \overline{E}_0}{\overline{E} - \underline{E}} < \frac{\pi}{2\Delta t}$$

Adding these inequalities and multiplying through by Δt, we see that (B.5) is satisfied. In the above derivation, β_0 canceled out entirely; the purpose of β_0 is to ensure that the rescaled eigenvalues satisfy $E_j \in [-\pi/(4\Delta t), \pi/(4\Delta t)]$. To see this, note that the above definitions and (B.6) yield

$$\begin{aligned} E_0 &= \frac{\pi}{2\Delta t \Delta E}[-\underline{E}/2 - \overline{E}/2 + \overline{E}_0] \\ &> \frac{\pi}{2\Delta t \Delta E}[-\underline{E}/2 - \overline{E}/2 + \underline{E}] = \frac{\pi}{2\Delta t \Delta E}\left[-\frac{\Delta E}{2}\right] = -\frac{\pi}{4\Delta t} \end{aligned}$$

Analogously, we can show $E_{N-1} < \pi/(4\Delta t)$. This together with the nondecreasing nature of the eigenvalues implies that all rescaled eigenvalues are in the desired interval. When $\Delta t = 1.0$, as it is in this chapter, we have $E_j \in [-\pi/4, \pi/4]$ for all j.

B.4 Supplementary Figures

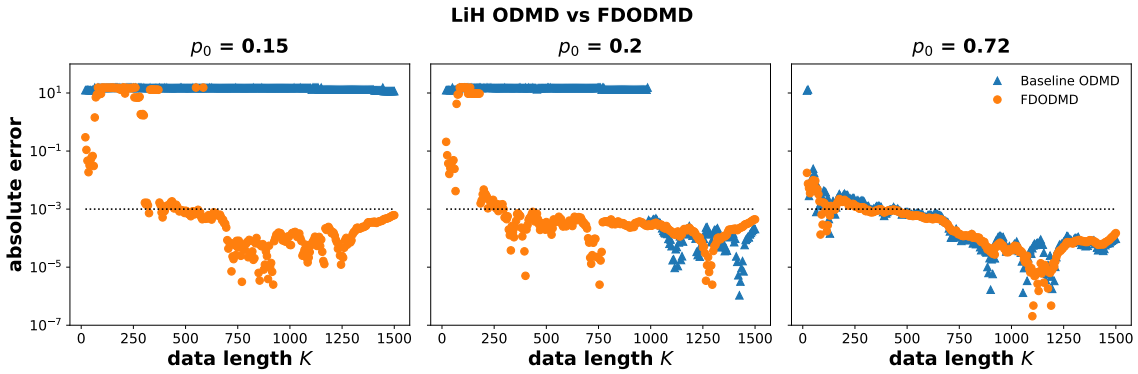


Figure B.3: Convergence comparison of FDODMD (circles) and baseline ODMD (triangles) for LiH. We choose $K_{\max} = 1500$, $\Delta t = 1.0$, and $p_0 \in \{0.15, 0.2, 0.72\}$. Both methods process noisy data given by (4.6) with $\xi(t_k) \sim \mathcal{N}(0, \epsilon = 0.1)$. For FDODMD, we use (4.17) with $\gamma \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5\}$ to generate 6 denoised realizations. All the denoised realizations are then stacked in conjunction with the noisy data according to (4.18). For both FDODMD and ODMD, we set $\delta = \epsilon$. Note the accelerated convergence (roughly 4 times less data required) to chemical accuracy offered by FDODMD for the lower overlap cases.

Fig. B.3 compares convergence to the GSE of another test molecule, LiH in the 3-21g basis, under the same parameter setting as in Fig. 4.5. We investigate the effect of varying

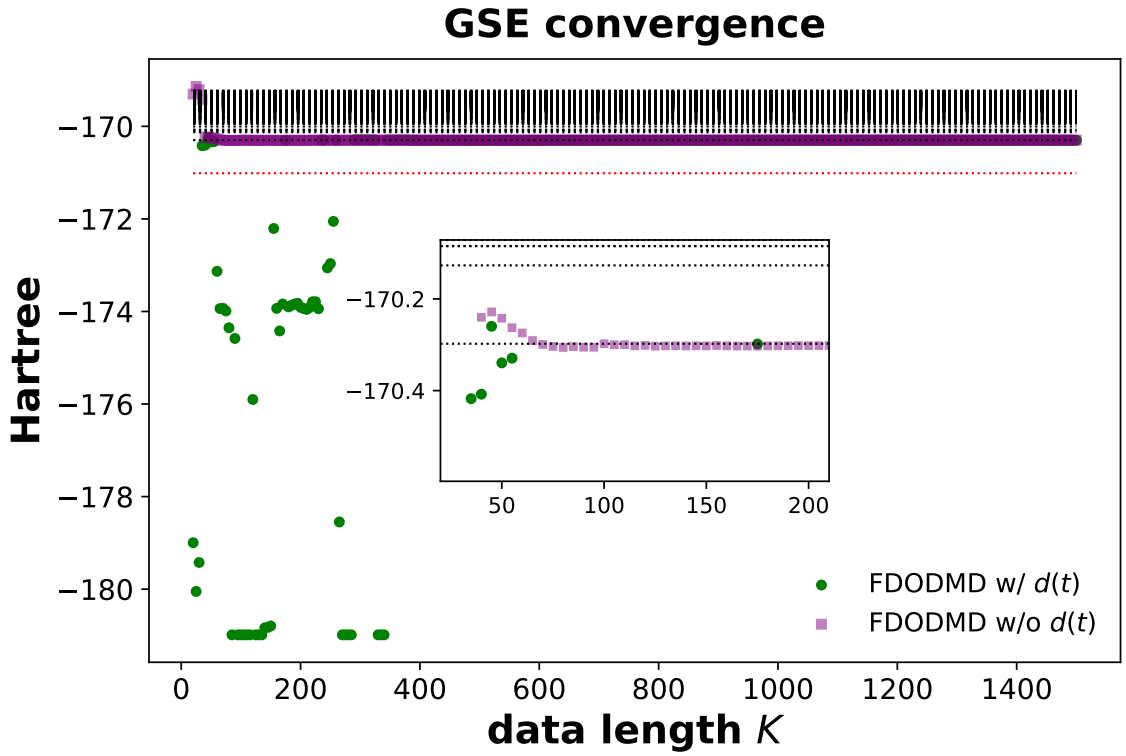
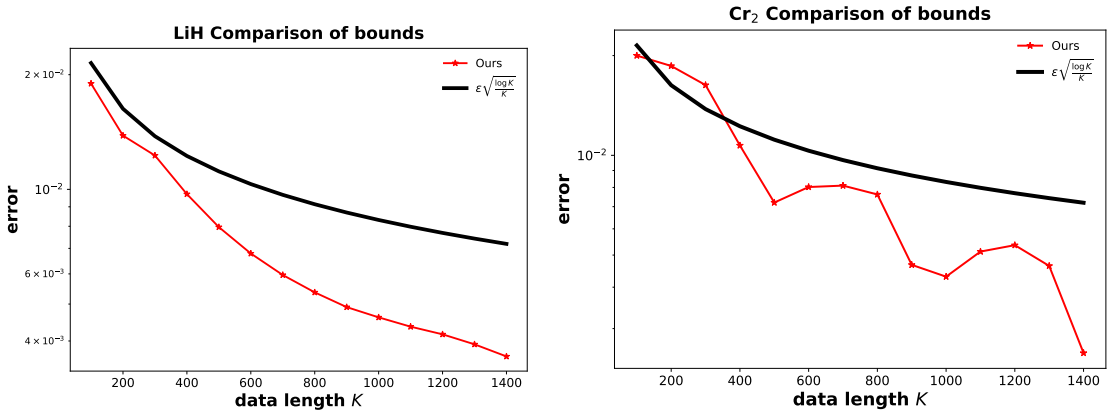


Figure B.4: Eigenvalue convergence of FDODMD with and without stacking the raw noisy data for Cr_2 molecule. Dashed lines indicate the target eigenenergies of H. We fix $(K_{\max}, \Delta t, p_0) = (1500, 1.0, 0.2)$ and simulate noisy data given by (4.6) with $\xi(t_k) \sim \mathcal{N}(0, \epsilon = 0.8)$. We choose $\delta = 0.8$ accordingly. For FDODMD, we use (4.17) with $\gamma \in \{2.0, 2.5, 3.0, 3.5, 4.0, 4.5\}$ to generate $R = 6$ denoised realizations. The red dashed line indicates the lower bound computed from Ref. [136]. When stacking includes the raw data in the high noise regime, the estimated ground state energy exhibits a clear violation of known lower bounds.



(a) LiH (b) Cr_2

Figure B.5: Comparison of the bound developed in (4.27) for both LiH and Cr_2 averaged across 100 noisy trajectories compared to the bound in Ref. [10]. For our bound, we choose the optimal truncation factor τ_r consistent with Fig. 4.12. Here, for both molecular systems we fix $\epsilon = 0.1$, $p_0 = 0.2$, and $\Delta t = 1.0$ to simulate noisy data from (4.6). We vary the length of the time series K for both bounds. Numerically, our bound appears tighter for these experiments.

the initial state overlap p_0. Similar to Cr_2, we find that FDODMD offers rapid and stable convergence compared to ODMD, requiring roughly 4 times less quantum resources.

Fig. B.4 examines the convergence of the FDODMD GSE estimate under the same parameter set used in Fig. 4.8. We compare the effect of stacking with and without the raw noisy data $d(t)$ from (4.3) in the high noise regime. Specifically, we evaluate two stacking strategies:

1. Stacking the raw noisy observable $d(t)$ together with its R denoised reconstructions.

2. Stacking only the R denoised trajectories.

As the denoising and stacking procedures are purely classical, both variants can be evaluated with minor cost by running the post-processing twice. When the noisy data is retained in the stack, the estimated GSE, as a function of the data length, manifestly violates known lower bounds reported in the literature [136, 120]. By contrast, the stack composed solely of the denoised signals respects both the lower and upper bounds over the observed time window.

Fig. B.5 compares the error bound derived in (4.27) with an alternative denoising bound based on the framework of atomic norm minimization [10]. In particular, Ref. [10]

shows that, under a canonically chosen denoising parameter (which differs from the Fourier mode thresholding considered in our approach), the mean-squared reconstruction error scales asymptotically as $\epsilon\sqrt{\log(K)/K}$. Thus, we fix the threshold τ_r at the optimum identified in Fig. 4.12 and plot both bounds as functions of K. Numerically, our bound yields favorable reductions in the reconstruction error over the entire range of K considered, supporting the performance gain of FDODMD over baseline ODMD shown in Section 3.5.

B.5 Formal Proof of Theorem 4.8.1

To recap, the noiseless, noisy, and denoised data are defined by (4.2), (4.3), and (4.17), respectively; $\mathbf{s}$, $\mathbf{d}$ and $\mathbf{r}$ denote their vectorized time domain representations. Given a complex-valued vector $\mathbf{x} = \{x_k\}_{k=0}^{K-1}$, we define the forward and inverse discrete Fourier transform (DFT) pair:

$$\widehat{x}_k = \sum_{j=0}^{K-1} x_j \, e^{-2\pi i j k/K} \quad \text{and} \quad x_k = \frac{1}{K} \sum_{j=0}^{K-1} \widehat{x}_j \, e^{2\pi i j k/K}. \tag{B.7}$$

For any vector $\mathbf{x} \in \mathbb{C}^K$, Plancherel's theorem states that

$$\|\mathbf{x}\|_2^2 = \frac{1}{K}\|\widehat{\mathbf{x}}\|_2^2,$$

where $\|\mathbf{x}\|_2 = (\mathbf{x}^\dagger \mathbf{x})^{1/2} = \left(\sum_{k=0}^{K-1} |x_k|^2\right)^{1/2}$ is the vector 2-norm with $(\cdot)^\dagger$ denoting conjugate transpose (while we will use $(\cdot)^T$ for ordinary transpose). By Plancherel's theorem,

$$\begin{aligned}
\|\mathbf{r} - \mathbf{s}\|_2^2 = \frac{1}{K}\|\widehat{\mathbf{r}} - \widehat{\mathbf{s}}\|_2^2 &= \frac{1}{K} \sum_{k=0}^{K-1} |\widehat{r}_k - \widehat{s}_k|^2 \\
&= \frac{1}{K} \sum_{k=0}^{K-1} \left| \mathbb{I}_{|\widehat{d}_k| \geq \tau_r} \widehat{d}_k - \widehat{s}_k \right|^2 \\
&= \frac{1}{K} \sum_{k=0}^{K-1} \left| \mathbb{I}_{|\widehat{d}_k| \geq \tau_r} (\widehat{s}_k + \widehat{\xi}_k) - \widehat{s}_k \right|^2 \\
&= \frac{1}{K} \sum_{k=0}^{K-1} \mathbb{I}_{|\widehat{d}_k| \geq \tau_r} |\widehat{\xi}_k|^2 + \mathbb{I}_{|\widehat{d}_k| < \tau_r} |\widehat{s}_k|^2.
\end{aligned} \tag{B.8}$$

To compute the expected value of the right-hand side, we must determine the distribution of $|\widehat{\xi}_k|^2$. The remainder of the proof proceeds by treating the noise in Fourier domain as a complex Gaussian process, and using properties of the exponential and error function distributions to compute the expected error.

To determine the distributions of the random vector $\boldsymbol{\xi}$ and its DFT $\widehat{\boldsymbol{\xi}}$, let $\mathsf{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the multivariate normal with PDF

$$f(\mathbf{w}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right)}{(2\pi)^{K/2}\det(\boldsymbol{\Sigma})^{1/2}}.$$

Based on our noise assumptions in Section 4.2.2, we have

$$\operatorname{Re}\boldsymbol{\xi} \sim \mathsf{N}(\mathbf{0}, \boldsymbol{\Sigma}) \;\text{ and }\; \operatorname{Im}\boldsymbol{\xi} \sim \mathsf{N}(\mathbf{0}, \boldsymbol{\Sigma}), \tag{B.9}$$

with $\boldsymbol{\Sigma} = \epsilon^2\mathbf{I}$. Let $\mathsf{CN}(\boldsymbol{\mu}, \boldsymbol{\Gamma}, \mathrm{C})$ denote the complex normal with PDF

$$g(\mathbf{z}) = \frac{1}{\pi^K\sqrt{\det(\boldsymbol{\Gamma})\det(\overline{\boldsymbol{\Gamma}} - \mathbf{C}^\dagger\boldsymbol{\Gamma}^{-1}\mathbf{C})}} \cdot \exp\left\{-\frac{1}{2}\begin{bmatrix}\mathbf{z}-\boldsymbol{\mu}\\ \overline{\mathbf{z}}-\overline{\boldsymbol{\mu}}\end{bmatrix}^\dagger \begin{bmatrix}\boldsymbol{\Gamma} & \mathbf{C}\\ \overline{\mathbf{C}} & \overline{\boldsymbol{\Gamma}}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{z}-\boldsymbol{\mu}\\ \overline{\mathbf{z}}-\overline{\boldsymbol{\mu}}\end{bmatrix}\right\}.$$

Combining (B.9), statistical independence of $\operatorname{Re}\boldsymbol{\xi}$ and $\operatorname{Im}\boldsymbol{\xi}$, and properties relating the multivariate and complex normal distributions, we conclude that

$$\boldsymbol{\xi} \sim \mathsf{CN}(\mathbf{0}, \boldsymbol{\Gamma}, \mathbf{0}), \tag{B.10}$$

with $\boldsymbol{\Gamma} = 2\boldsymbol{\Sigma} = 2\epsilon^2\mathbf{I}$. Now we define the forward Fourier matrix $\mathcal{F}$ via

$$\mathcal{F}_{jk} = e^{-2\pi ijk/K},$$

for $0 \le j, k \le K-1$. Note the use of zero-based indexing for row/column entries. With this, the forward and inverse DFT (B.7) can be written as a matrix-vector multiplications,

$$\widehat{\boldsymbol{\xi}} = \mathcal{F}\boldsymbol{\xi} \quad \text{and} \quad \boldsymbol{\xi} = \frac{1}{K}\mathcal{F}^\dagger\widehat{\boldsymbol{\xi}}. \tag{B.11}$$

Hence $\mathcal{F}^\dagger\mathcal{F} = \mathcal{F}\mathcal{F}^\dagger = K\mathbf{I}$. Modulo the normalization constant K, the matrix $\mathcal{F}$ is unitary. Combining (B.11) with (B.10), we have

$$\widehat{\boldsymbol{\xi}} \sim \mathsf{CN}(\mathbf{0}, \mathcal{F}\boldsymbol{\Gamma}\mathcal{F}^\dagger, \mathbf{0}).$$

where $\mathcal{F}\boldsymbol{\Gamma}\mathcal{F}^\dagger = 2\epsilon^2\mathcal{F}\mathcal{F}^\dagger = 2\epsilon^2 K\mathbf{I}$. Hence

$$\mathrm{Re}\,\widehat{\boldsymbol{\xi}} \sim \mathsf{N}(\mathbf{0}, \epsilon^2 K\mathbf{I}) \;\text{ and }\; \mathrm{Im}\,\widehat{\boldsymbol{\xi}} \sim \mathsf{N}(\mathbf{0}, \epsilon^2 K\mathbf{I}),$$

which means that each component is independent and has the same univariate normal distribution with mean 0 and variance $\epsilon^2 K$. Standardizing, we have

$$\epsilon^{-1}K^{-1/2}\,\mathrm{Re}\,\widehat{\xi}_k, \epsilon^{-1}K^{-1/2}\,\mathrm{Im}\,\widehat{\xi}_k \sim \mathcal{N}(0,1).$$

The sum of squares of two independent standard normals,

$$W = (\epsilon^{-1}K^{-1/2}\,\mathrm{Re}\,\widehat{\xi}_k)^2 + (\epsilon^{-1}K^{-1/2}\,\mathrm{Im}\,\widehat{\xi}_k)^2 = \epsilon^{-2}K^{-1}|\widehat{\xi}_k|^2,$$

has an exponential distribution with rate parameter $1/2$. With this, we can conclude that $|\widehat{\xi}_k|^2$, which appears in the first summation of (B.8), is exponentially distributed with parameter $\lambda = \epsilon^{-2}K^{-1}/2$. This implies that

$$\mathbb{E}[|\widehat{\xi}_k|^2] = \lambda^{-1} = 2K\epsilon^2.$$

Summing this over k, we obtain $\mathbb{E}[\|\widehat{\boldsymbol{\xi}}\|_2^2] = 2K^2\epsilon^2$: Plancherel's theorem implies $\mathbb{E}[\|\boldsymbol{\xi}\|_2^2] = 2K\epsilon^2$, which is indeed consistent with (B.10). We also conclude from the above that $|\widehat{\xi}_k|$ has a Rayleigh distribution with parameter $\sigma = 1/\sqrt{2\lambda} = \epsilon K^{1/2}$.

On the other hand, we note that $|\widehat{d}_k| = |\widehat{s}_k + \widehat{\xi}_k| \le |\widehat{s}_k| + |\widehat{\xi}_k|$, and $\{k\,||\widehat{d}_k| \ge \tau_r\} \subset \{k\,||\widehat{s}_k| + |\widehat{\xi}_k| \ge \tau_r\}$. Using this, we can take the expected value of the first term in (B.8) and derive

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{I}_{|\widehat{d}_k|\ge\tau_r}|\widehat{\xi}_k|^2\right] \le \frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\mathbb{I}_{|\widehat{\xi}_k|\ge\tau_r-|\widehat{s}_k|}|\widehat{\xi}_k|^2\right].$$

There are two cases. When $\tau_r < |\widehat{s}_k|$, the indicator function on the right-hand side always remains 1, so the expected value on the right-hand side is $2K\epsilon^2$ as derived above. When $\tau_r \ge |\widehat{s}_k|$,

$$\mathbb{E}\left[\mathbb{I}_{|\widehat{\xi}_k|\ge\tau_r-|\widehat{s}_k|}|\widehat{\xi}_k|^2\right] = \mathbb{E}\left[\mathbb{I}_{|\widehat{\xi}_k|^2\ge(\tau_r-|\widehat{s}_k|)^2}|\widehat{\xi}_k|^2\right].$$

As we know the distribution of $|\widehat{\xi}_k|^2$, we can directly evaluate the expectation: for $\alpha \ge 0$,

$$\mathbb{E}\left[\mathbb{I}_{|\widehat{\xi}_k|^2\ge\alpha}|\widehat{\xi}_k|^2\right] = \int_\alpha^\infty w\lambda e^{-\lambda w}\,dw = \frac{e^{-\alpha\lambda}}{\lambda}(1+\alpha\lambda).$$

For convenience, we define the function $q(z) = e^{-z}(1+z)$, in terms of which we have

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{I}_{|\widehat{d}_k|\geq\tau_r}|\widehat{\xi}_k|^2\right] \leq 2K\epsilon^2\left[\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{I}_{\tau_r\geq|\widehat{s}_k|}q\left(\frac{(\tau_r-|\widehat{s}_k|)^2}{2\epsilon^2K}\right)+\mathbb{I}_{\tau_r<|\widehat{s}_k|}\right]. \quad \text{(B.12)}$$

We now return to the second half of (B.8). Taking expected values, we have

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{I}_{|\widehat{d}_k|<\tau_r}|\widehat{s}_k|^2\right] = \frac{1}{K}\sum_{k=0}^{K-1}|\widehat{s}_k|^2\mathbb{P}\Big(\Big|\widehat{s}_k+\widehat{\xi}_k\Big|<\tau_r\Big). \quad \text{(B.13)}$$

Note that

$$\mathbb{P}\Big(\Big|\widehat{s}_k+\widehat{\xi}_k\Big|<\tau_r\Big) = \int_{\Omega_k}\frac{1}{2\pi\epsilon^2K}\exp\left(-\frac{x^2+y^2}{2\epsilon^2K}\right)dx\,dy,$$

where $\Omega_k = \{(x,y)\in\mathbb{R}^2\,|\,(\operatorname{Re}\widehat{s}_k+x)^2+(\operatorname{Im}\widehat{s}_k+y)^2<\tau_r^2\}$ is the disk centered at $(-\operatorname{Re}\widehat{s}_k,-\operatorname{Im}\widehat{s}_k)$ with diameter $2\tau_r$. Clearly, $\Omega_k\subset S_k$ for a square S_k with the same center and side length $2\tau_r$. Therefore,

$$\begin{aligned}\mathbb{P}\Big(\Big|\widehat{s}_k+\widehat{\xi}_k\Big|<\tau_r\Big) &\leq \int_{S_k}\frac{1}{2\pi\epsilon^2K}\exp\left(-\frac{x^2+y^2}{2\epsilon^2K}\right)dx\,dy\\ &\leq \frac{1}{4}\left[\operatorname{Erf}\left(\frac{-\operatorname{Re}\widehat{s}_k+\tau_r}{\sqrt{2\epsilon^2K}}\right)-\operatorname{Erf}\left(\frac{-\operatorname{Re}\widehat{s}_k-\tau_r}{\sqrt{2\epsilon^2K}}\right)\right]\\ &\quad\cdot\left[\operatorname{Erf}\left(\frac{-\operatorname{Im}\widehat{s}_k+\tau_r}{\sqrt{2\epsilon^2K}}\right)-\operatorname{Erf}\left(\frac{-\operatorname{Im}\widehat{s}_k-\tau_r}{\sqrt{2\epsilon^2K}}\right)\right], \quad \text{(B.14)}\end{aligned}$$

where $\operatorname{Erf}(z) = 2/\sqrt{\pi}\int_0^z e^{-t^2}\,dt$ denotes the error function. Let us define

$$\begin{aligned}\mathcal{R}_k(\tau_r,K,\epsilon) := &\left[\operatorname{Erf}\left(\frac{-\operatorname{Re}\widehat{s}_k+\tau_r}{\sqrt{2\epsilon^2K}}\right)-\operatorname{Erf}\left(\frac{-\operatorname{Re}\widehat{s}_k-\tau_r}{\sqrt{2\epsilon^2K}}\right)\right]\\ &\cdot\left[\operatorname{Erf}\left(\frac{-\operatorname{Im}\widehat{s}_k+\tau_r}{\sqrt{2\epsilon^2K}}\right)-\operatorname{Erf}\left(\frac{-\operatorname{Im}\widehat{s}_k-\tau_r}{\sqrt{2\epsilon^2K}}\right)\right]. \quad \text{(B.15)}\end{aligned}$$

Combining (B.14) with (B.13) and (B.12), we can estimate the expected value of the RHS

of (B.8), resulting in

$$\frac{1}{K}\,\mathbb{E}\left[\|\mathbf{r}-\mathbf{s}\|_2^2\right] \leq 2\epsilon^2\left[\frac{1}{K}\sum_{k=0}^{K-1}\left(\mathbb{I}_{\{\tau_r\geq|\widehat{s}_k|\}}\,q\Big(\frac{(\tau_r-|\widehat{s}_k|)^2}{2\epsilon^2K}\Big)+\mathbb{I}_{\{\tau_r<|\widehat{s}_k|\}}\right)\right] + \frac{1}{K}\sum_{k=0}^{K-1}\frac{|\widehat{s}_k|^2}{4K}\,\mathcal{R}_k(\tau_r,K,\epsilon). \quad \text{(B.16)}$$

where we can identify the RHS with the function $G(\tau_r, K, \epsilon, \mathbf{s})$ in Theorem 4.8.1. □

With further analysis, we can unpack the K-dependence of the right-hand side of (B.16). The first bracketed term is a K-normalized average of bounded factors and is therefore $O(1)$ in K (for fixed τ_r). Via a mean-value theorem argument, we show below that the second term is $O(1/K^2)$, after summation and application of Plancherel's theorem. Hence, the right-hand side of (B.16) equals a τ_r–controlled constant plus an $O(1/K^2)$ term.

For any $u \in \mathbb{R}$ and $\delta > 0$, the mean value theorem gives

$$\mathrm{Erf}(u+\delta)-\mathrm{Erf}(u-\delta) = 2\delta\,\mathrm{Erf}'(u^*),$$

for some $u^* \in (u-\delta, u+\delta)$. Since $\mathrm{Erf}'(x) = 2/\sqrt{\pi}e^{-x^2} \leq 2/\sqrt{\pi}$ for all x,

$$\Big|\mathrm{Erf}(u+\delta)-\mathrm{Erf}(u-\delta)\Big| \leq \frac{4\delta}{\sqrt{\pi}}. \quad \text{(B.17)}$$

Apply (B.17) with

$$u_x = -\frac{\mathrm{Re}\,\widehat{s}_k}{\sqrt{2\epsilon^2K}}, \qquad u_y = -\frac{\mathrm{Im}\,\widehat{s}_k}{\sqrt{2\epsilon^2K}}, \qquad \delta = \frac{\tau_r}{\sqrt{2\epsilon^2K}}.$$

Then, each bracket in $\mathcal{R}_k$ (see (B.15)) satisfies

$$\left|\mathrm{Erf}\left(\frac{-\mathrm{Re}\,\widehat{s}_k+\tau_r}{\sqrt{2\epsilon^2K}}\right)-\mathrm{Erf}\left(\frac{-\mathrm{Re}\,\widehat{s}_k-\tau_r}{\sqrt{2\epsilon^2K}}\right)\right| \leq \frac{2\sqrt{2}}{\sqrt{\pi}}\cdot\frac{\tau_r}{\epsilon\sqrt{K}},$$

and the same bound holds with Re replaced by Im. Hence

$$\mathcal{R}_k(\tau_r,K,\epsilon) \leq \left(\frac{2\sqrt{2}}{\sqrt{\pi}}\cdot\frac{\tau_r}{\epsilon\sqrt{K}}\right)^2 = \frac{8\tau_r^2}{\pi\epsilon^2K}. \quad \text{(B.18)}$$

Note that the shifts $-\operatorname{Re}\widehat{s}_k$ and $-\operatorname{Im}\widehat{s}_k$ only change u^* in the mean value theorem; the bound (B.17) is uniform in u.

Inserting (B.18) into the second term of (B.16), and then applying Plancherel, we have:

$$\frac{1}{K}\sum_{k=0}^{K-1}\frac{|\widehat{s}_k|^2}{4K}\,\mathcal{R}_k(\tau_r,K,\epsilon)\le\frac{2\tau_r^2}{\pi\epsilon^2K^3}\sum_{k=0}^{K-1}|\widehat{s}_k|^2\le\frac{2\tau_r^2}{\pi\epsilon^2}\,\|\mathbf{s}\|_2^2\cdot\frac{1}{K^2}.$$

In the last expression, the constant multiplying $1/K^2$ depends only on τ_r, ϵ, and $\mathbf{s}$.

Let us return to (B.16). Since $q(\cdot)\in(0,1]$ and the indicators are in $\{0,1\}$, the first bracketed term on the right-hand side of (B.16) is $O(1)$ in K, with the precise constant controlled by the threshold τ_r. The second term is $O(1/K^2)$ by the bound above. Thus, the mean-squared error bound of Theorem 4.8.1 can be interpreted as follows:

$$\frac{1}{K}\,\mathbb{E}\Big[\|\mathbf{r}-\mathbf{s}\|_2^2\Big]\quad\le\quad 2\epsilon^2\Big[\text{threshold-controlled constant}\Big]\;+\;\left(\frac{2}{\pi}\,\frac{\tau_r^2}{\epsilon^2}\,\|\mathbf{s}\|_2^2\right)\;\cdot\;\frac{1}{K^2},$$