A NOVEL FRAMEWORK FOR MACHINE LEARNING: PARAMETRIC MATRIX MODELS AND ITS APPLICATION IN NUCLEAR PHYSICS, SCIENTIFIC COMPUTING, AND GENERAL AI.

By

Danny Jammooa

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Physics—Doctor of Philosophy
Computational Mathematics, Sciences and Engineering—Dual Major

2025

ABSTRACT

Complex nuclear systems present formidable challenges, including the exponential growth of Hilbert-space dimensions, the slow convergence of perturbative expansions, and the difficulty of extracting excited-state information from limited-order data. Addressing these intricate many-body interactions and the need for highly accurate and interpretable computational frameworks necessitates innovative algorithmic solutions that can efficiently learn governing equations and provide robust predictions.

This thesis introduces and extensively develops Parametric Matrix Models (PMMs), a novel class of machine learning algorithms specifically designed to address these challenges. PMMs fundamentally differ from conventional neural network and deep learning inspired approaches by emulating physical systems through matrix equations. This design allows PMMs to learn the underlying governing equations directly from empirical data, offering an interpretable and efficient computational framework capable of input feature extrapolation.

Building upon the foundational theory of PMMs, this research significantly extends their applicability. We generalize PMMs to effectively model complex physical phenomena, including state-vector evolution and nonlinear dynamics. The primary focus of this work lies in the application of PMMs to imaginary time evolution data within Quantum Monte Carlo lattice Effective Field Theory (EFT). Here, PMMs are demonstrated to accurately resum perturbation theory from first-order approximations, enabling robust predictions for experimental observables such as ground state energies, charge radii, and first excited states in nuclear systems. Furthermore, this thesis explores the versatility of PMMs by presenting a tailored Navier-Stokes PMM, designed to learn and analyze data from Computational Fluid Dynamics (CFD) simulations, their application to general machine learning problems, including regression and image classification.

By leveraging the inherent mathematical structures of physics, this approach offers a powerful and innovative algorithmic solution. This work underscores the transformative potential of physics-inspired machine learning for advancing scientific discovery and addressing critical challenges across a wide range of scientific and general machine learning problems.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank all of the people who have supported me throughout my Ph.D. journey. Their guidance, encouragement, and friendship have made this work possible and memorable.

I would first like to thank my co-advisors, Dean Lee and Morten Hjorth-Jensen, for their invaluable guidance and support throughout my Ph.D. journey. To Dean, I am especially grateful for encouraging me to attend conferences and present my work, for always being available for discussions, and for engaging in countless conversations about the many possible directions for future research. Beyond research, I deeply appreciate how you have always looked out for me, offering guidance on postdoctoral opportunities and helping me navigate the next steps in my career. Your support and mentorship have been a constant source of motivation and confidence throughout my Ph.D. To Morten, I am deeply thankful for inviting me to spend time in Oslo, for encouraging me to attend ECT* in Italy, which offered me a new perspective. Beyond research, I greatly appreciated your hospitality, inviting me to your home, showing me the local area, hiking in your backyard in Oslo, and taking me to a beautiful beach cove in Italy. Additionally, given your background in education, you taught me the value of writing clear, pedagogical notebooks when collaborating with others. I am also grateful to both of you for your mentorship even before the start of my Ph.D., when I worked with you during my gap year between undergraduate and graduate studies. That experience gave me my first introduction to research and allowed me to explore machine learning through the FRIB summer school, which shaped much of the path I would later follow.

I would also like to thank my co–first author on my very first publication, Patrick Cook. Ever since we met in Oslo, I am grateful that I was able to convince you to join me in this research. I knew that working together would only elevate it to what it has become today. I truly appreciated how you were always willing to discuss our work, the many productive back-and-forth exchanges we shared, and the new perspectives you brought. Beyond research, I also enjoyed our many car rides and ridiculous conversations, from summiting the mountain in Hawaii to our trips to and from

Argonne National Laboratory.

Next, I would like to thank my office mates, Manuel Catacorarios and Jacob Watkins, for all the countless random arguments we found ourselves in over the most unexpected topics. To Manuel, whom I first met back in undergrad when we took Electricity and Magnetism together, I still remember you convincing me to skip undergraduate quantum mechanics and jump straight into the graduate course, a decision I regret to this day! Nevertheless, it has been a lot of fun going through both our undergraduate and graduate journeys side by side.

Following that, I would like to thank the rest of my group members, Nick Cariello and Yuanzhuo Ma. To Nick, I greatly enjoyed our morning conversations and working through the New York Times puzzles together, as well as your constant quest to find the best food whenever we traveled for conferences. To Yuanzhuo, although I only got to know you more recently, I appreciate all of our conversations and am glad that we have been working together on this new project. I hope that our collaboration continues even after graduation.

I would also like to thank former group members Jane Kim and Julie Hartley for their help and support early in my grad career, and for staying in touch over the years at conferences. I'm also grateful to Julie and Adam Hartley for the time we spent together in Oslo, especially trying out the local food and our advisor's favorite drink.

I would also like to thank Michael Serikow and Joseph Dopfer, whom I got to know during graduate school. Getting together each semester to catch some food or a movie was always a great way to unwind and balance out research life.

I would also like to thank Erick Flynn, Daniel Lay, Brandon Lem, Michael Gajdosik, Jordan Purcell, Ben Clark, Shane Blade, Alexandra Semposki, Sudhanva Lalit, and Kang Yu, from the lab at FRIB and beyond, whom I have had the pleasure of getting to know over the years. I am equally grateful to Daniel Lee, Nick Rohde, Elisha Alemao, undergraduate students, I have had the chance to mentor. I would also like to thank my committee members, Michael Murillo, Alexi Bazavov, and Jonas Becker, for their time, thoughtful feedback, and guidance throughout this process. In addition, I am grateful to the many professors and FRIB/MSU faculty I enjoyed talking to over the

years, including Heiko Hergert, Witold Nazarewicz, Scott Pratt, Xilin Zhang, Pablo Giuliani, Kyle Godbey, Ryan Larose, Piotr Piecuch, and Danny Caballero whose conversations and insights helped broaden my perspective. With a special mention to Elizabeth Deliyski and Kimberly Crosslan, who were always open to chat and generous with their help, especially when it came to navigating travel and FRIB/MSU-related issues. Finally, I would like to thank FRIB and Michigan State University for providing the resources, community, and opportunities that made this work possible.

To my family and friends, thank you for your constant encouragement, patience, and belief in me throughout this journey. I am especially grateful to my parents for their unwavering support, guidance, and financial assistance, which made this work possible. Your encouragement has given me the freedom to pursue my goals.

Finally, I'd like to thank my cat, Ori, for keeping me company through long nights of work, and my brothers' cat, Bubbles, who also deserves a mention for occasionally being awesome.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

PMMs Parametric Matrix Models

NN Neural Network

MLP Multilayer Perceptron

CNN Convolutional Neural Networks

PINNs Physics Informed Neural Networks

POD Proper Orthogonal Decomposition

EVD Eigenvalue Decomposition

SVD Singular Value Decomposition

TN Tensor Networks

MPS Matrix Product States

DMRG Density Matrix Renormalization Group

TEBD Time-Evolving Block Decimation

NQS Neural Quantum States

CFD Computational Fluid Dynamics

LMG Lipkin-Meshkov-Glick

ALMG Anharmonic Lipkin-Meshkov-Glick

CC Couple Cluster

IMSRG In Medium Similarity Renormalization Group

EC Error Correction

NS Navier-Stokes

EFT Effective Field Theory

χ**EFT** Chiral Effective Field Theory

ML Machine Learning

AI Artificial Intelligence

ROM	Reduced Order Model
DMD	Dynamic Mode Decomposition
QC	Quantum Computing
QCD	Quantum Chromodynamics
MBPT	Many-Body Perturbation Theory
SCGF	Self-Consistent Green's Function
MC	Monte Carlo
QMC	Quantum Monte Carlo
AFDMC	Auxiliary-Field Diffusion Monte Carlo
GFMC	Green's Function Monte Carlo
VQE	Variational Quantum Eigensolver
QPE	Quantum Phase estimation
NISQ	Noisy Intermediate-Scale Quantum
ZNE	Zero-Noise Extrapolation
OLS	Ordinary Least Squares
TTN	Tree Tensor Networks
MERA	Multiscale Entanglement Renormolization Ansatz
PEPs	Projected Entangled Pair States
RBM	Reduced Basis Method
Qubit	Quantum Bit
QITE	Quantum Imaginary- Time Evolution

CHAPTER 1

INTRODUCTION

1.1 Overview & Motivation

The pursuit of understanding complex physical systems, particularly within nuclear physics and the broader realm of quantum mechanics, presents formidable computational and theoretical challenges. These include, but are not limited to, the exponential growth of Hilbert-space dimensions, which makes direct computation intractable; the notoriously slow convergence of perturbative expansions, limiting the accuracy and applicability of theoretical models; and the inherent difficulty of extracting crucial excited-state information from limited-order experimental or theoretical data. Addressing these intricate many-body interactions and the need for highly accurate, interpretable, and extrapolatable computational frameworks necessitates the development of innovative algorithmic solutions.

Recognizing this critical need, artificial intelligence (AI) and machine learning (ML) have emerged as transformative tools, enhancing the ability to perform advanced physics calculations and tackle the intricacies of complex nuclear systems. This potential has been acknowledged at the highest levels, exemplified by the US government launched a comprehensive, multidisciplinary, multi-agency program in 2016 aimed at establishing a sustained national AI infrastructure. In alignment with these strategic priorities, the Department of Energy (DOE) Office of Nuclear Physics has taken significant steps to integrate AI and ML into its research agenda. This includes hosting specialized events such as the "AI for Nuclear Physics Workshop" in March 2020 and the "SC Roundtable on Transformational Science Enabled by Artificial Intelligence: High Energy and Nuclear Physics" in October 2024, both designed to explore scientific opportunities and address the challenges inherent in applying AI/ML to nuclear physics and high-performance computing. Additionally, the Nuclear Science Advisory Committee's Long Range Plan [67] emphasizes the importance of promoting research in AI/ML for nuclear theory, further reinforcing the commitment to innovation and collaborative progress in this field.

Against this backdrop, this thesis introduces and extensively develops Parametric Matrix Models

(PMMs), a novel class of machine learning algorithms specifically designed to address these profound challenges. Unlike most existing machine learning models that imitate the biology of neurons, PMMs fundamentally diverge by employing matrix equations that directly emulate physical systems. This unique design allows PMMs to learn the underlying governing equations from empirical data, offering an inherently interpretable and efficient computational framework capable of robust input feature extrapolation. This work demonstrates how PMMs provide a powerful and innovative algorithmic solution by leveraging the inherent mathematical structures of physics, ultimately advancing scientific discovery and addressing critical challenges across a wide range of scientific and general machine learning problems.

The remainder of this thesis is organized into several thematic sections. Chapters 2- 5 provide comprehensive overviews of the key fields relevant to this work, outlining their current challenges and motivating the development of new methodologies.

Chapter 2 focuses on nuclear physics. It begins with an exploration of the nuclear many-body problem and reviews theoretical techniques used to address it. The chapter then discusses the limitations of these methods, which underscore the urgent need for a novel machine learning algorithm tailored to overcome these challenges.

Chapter 3 introduces quantum computing. While it may initially appear tangential, this chapter highlights how quantum computing is emerging as a powerful tool for tackling complex problems in nuclear physics. Here, the discussion centers on the current limitations of quantum computing and explores how machine learning techniques can mitigate errors inherent in quantum systems.

Chapter 4 provides an overview of machine learning and optimization techniques. This chapter reviews key optimization algorithms and machine learning models that are generally used and currently applied in nuclear physics. It also argues for the development of new, domain-specific machine learning models that can more effectively address the unique challenges in nuclear physics, as well as a general machine learning framework rooted in quantum mechanics.

Chapter 5 introduces Tensor Networks (TNs), a framework first developed to study quantum many-body physics. TNs compress high-dimensional spaces into tractable forms through structured

factorizations, making them powerful tools for problems where direct methods fail. While rooted in physics, TNs have recently found applications in machine learning, where efficient contraction paths and their expressive structure provide new strategies for representation and optimization.

Following these foundational chapters, Chapter 6 presents the novel machine learning approach introduced in this thesis, parametric matrix models (PMMs). The chapter is divided into two main parts: first, it elaborates on the underlying theory of PMMs; second, it demonstrates the application of PMMs across a range of problems. The applications are organized progressively: starting with nuclear physics, moving on to computational fluid simulations, and finally extending to general regression and classification tasks.

Finally, Chapter **??** concludes the thesis by summarizing the key findings and outlining future research directions for PMMs and their potential impact on machine learning in nuclear physics.

CHAPTER 2

NUCLEAR PHYSICS

2.1 Introduction

Nuclear physics stands at the forefront of humanity's quest to understand the fundamental building blocks of matter and the forces that govern their interactions. It delves into the intricate structure of atomic nuclei, the dynamics of nuclear reactions, and the properties of nuclear matter under extreme conditions, such as those found in neutron stars and supernovae. Insights gained from nuclear physics are not only crucial for comprehending the universe's evolution and the origin of elements but also have profound implications for diverse fields, including energy production, medical diagnostics and therapy, and national security.

However, simulating nuclear systems presents formidable challenges. The strong nuclear force, which binds protons and neutrons within the nucleus, is highly complex and non-perturbative at low energies. This complexity, coupled with the quantum mechanical nature of many-body systems, leads to significant computational hurdles. Accurately describing the interactions of multiple nucleons and predicting nuclear properties from first principles requires sophisticated theoretical frameworks and immense computational resources. These inherent difficulties necessitate the continuous development of innovative approaches to push the boundaries of our understanding.

2.2 Many-Body Physics

At the core of nuclear theory is the task of solving the many-body Schrödinger equations, both in their time-independent and time-dependent forms:

$$H|\psi^A\rangle = E|\psi^A\rangle \tag{2.1}$$

$$i\hbar\frac{d}{dt}|\psi^A(t)\rangle = H|\psi^A(t)\rangle \tag{2.2}$$

Here, $|\psi^A\rangle$ denotes an eigenstate of an A-body system with energy E, while $|\psi^A(t)\rangle$ represents the time-evolving quantum state. The objective of nuclear many-body theory is to compute accurate and systematically improvable solutions to these equations, starting from a microscopic Hamiltonian. This endeavor is crucial for understanding the fundamental properties of atomic nuclei, from their

binding energies and excitation spectra to their reactions and astrophysical implications.

The foundation of the many-body problem is the construction of the Hilbert space in which these equations are solved. The many-body Hilbert space is constructed from a one-body space $\mathcal{H}_1$, which contains all admissible single-particle wavefunctions. For instance, in three-dimensional space, $\mathcal{H}_1 \subset L^2\mathbb{R}^3$ known as the Lebesgue space of square-integrable functions. The full A-body space is then built as a tensor product of A such spaces:

$$\mathcal{H}_1^{\otimes A} = \mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_1 \tag{2.3}$$

However, due to the fundamental principles of quantum statistics, the physically relevant space for nucleons, which are fermions, is the antisymmetric subspace (for indistinguishable particles with half-integer spin). For bosons (particles with integer spin), it would be the symmetric subspace. This antisymmetrization (or symmetrization) drastically reduces the number of valid states compared to the full tensor product space. Despite this reduction, the dimension of the many-body Hilbert space still grows combinatorially with the number of particles A, posing a significant computational challenge.

The dynamics of the nuclear system are encoded in the nuclear Hamiltonian, which is typically modeled as a sum of one-body kinetic energy and multi-body interaction terms:

$$H = \sum_i T_i + \sum_{i<j} V_{ij} + \sum_{i<j<k} V_{ijk} \tag{2.4}$$

where T_i is the kinetic energy of the i-th nucleon, V_{ij} is a two-body potential, and V_{ijk} represents three-body interactions. While higher-body interactions (e.g., four-body) can exist, they are often neglected due to their smaller expected contribution and increased computational complexity. These interaction terms are systematically derived from chiral effective field theory (χEFT), which provides a low-energy realization of Quantum Chromodynamics (QCD) using nucleons and pions as degrees of freedom, while incorporating the symmetries of the underlying theory.

Despite such systematic frameworks, two major obstacles persist. First, the nuclear interaction cannot yet be derived exactly from QCD and must be modeled through effective theories like χEFT,

meaning the Hamiltonian itself is an approximation. Second, the exponential scaling of the many-body Hilbert space makes exact diagonalization infeasible for anything but light nuclei. Even the most complete methods, such as No-Core Full Configuration Interaction (FCI), scale exponentially with A, limiting their reach to small systems. As a result, advancing nuclear many-body physics depends critically on the development of methods that balance fidelity with tractability.

These methods, often referred to as ab initio methods, seek to solve the many-body Schrödinger equation directly from the fundamental nucleon-nucleon and three-nucleon interactions derived from QCD via χEFT, without relying on phenomenological parameters. The subsequent section will delve into the various ab initio approaches employed in modern nuclear physics, outlining their principles, strengths, and limitations.

2.3 Ab Initio Methods in Nuclear Physics

The inherent complexities of the nuclear many-body problem, as discussed in the preceding section, necessitate the development of sophisticated theoretical and computational approaches. *Ab initio* methods represent a class of such approaches that aim to solve the many-body Schrödinger equation directly from fundamental nucleon-nucleon (NN) and three-nucleon (3N) interactions, derived from the underlying theory of Quantum Chromodynamics (QCD) [39]. This section will first elaborate on the crucial role of chiral effective field theory (χEFT) in constructing these interactions, followed by an overview of prominent *ab initio* many-body methods, their essential role in modern nuclear physics, and the persistent challenges they face.

2.3.1 Chiral Effective Field Theory: The Foundation of *Ab Initio* Interactions

As QCD is non-perturbative at the low energies relevant for nuclear structure, a direct derivation of nuclear forces from quarks and gluons is currently intractable for many-body calculations. **Chiral Effective Field Theory (χEFT)** provides the essential bridge, offering a systematic and model-independent framework to construct nuclear forces consistent with the symmetries of QCD. It exploits the approximate chiral symmetry of QCD, which is spontaneously broken, leading to pions as pseudo-Goldstone bosons [63].

Within χEFT, the nuclear potential is organized as a power series in terms of a small expansion

parameter, typically denoted as Q/Λ_χ, where Q represents a low momentum or pion mass, and Λ is the breakdown scale of the theory [51, 39]. This expansion systematically includes contributions from pion exchanges (long-range forces) and contact interactions (short-range forces), along with their derivatives. The Hamiltonian can thus be written as:

$$H = \sum_i T_i + V_{NN} + V_{NNN} + \cdots = \sum_i T_i + \sum_n \left(\frac{Q}{\Lambda_\chi}\right)^n V^{(n)} \tag{2.5}$$

where $V^{(n)}$ denotes interaction terms appearing at order n in the chiral expansion. This systematic expansion allows for a controlled truncation and, crucially, provides a means to estimate theoretical uncertainties associated with the omitted higher-order terms. The low-energy constants (LECs) that appear at each order are determined by fitting to a minimal set of experimental data, typically few-nucleon scattering observables.

2.3.2 Overview of First-Principle Many-Body Approaches

Given a chiral EFT Hamiltonian, various *ab initio* many-body methods are employed to solve the Schrödinger equation. These methods differ in their theoretical formalism and computational scaling, falling into two categories: configuration space methods or coordinate-space methods that work directly with the wave function, each offering unique strengths and limitations [39].

Some approaches provide systematic, perturbative corrections to a reference state, offering clear hierarchies and uncertainty estimates but may break down in strongly correlated or near-degenerate regimes. Non-perturbative methods, by contrast, capture strong correlations more robustly at the cost of increased computational effort. Other techniques excel at extracting dynamical properties, spectral functions, or excited-state information, while Monte Carlo–based and lattice methods handle complex geometries, cluster structures, and continuum effects but can be limited by the sign problem or discretization artifacts. In practice, the choice of method depends on the type of observable, desired accuracy, and tractable system size, with different approaches often complementing one another in a comprehensive study of nuclear systems.

2.3.2.1 Many-Body Perturbation Theory (MBPT)

MBPT is the simplest configuration-space approach for systematically improving upon a simpler reference state (e.g., a Hartree-Fock or Brueckner-Hartree-Fock solution) [75]. The full Hamiltonian H is split into an unperturbed part H_0 and a perturbation V', such that $H = H_0 + \lambda V'$. The energy E and wavefunction $|\psi\rangle$ are then expanded in a perturbative series:

$$E_n(\lambda) = E_n^{(0)} + \lambda \langle \psi_n^{(0)} | V' | \psi_n^{(0)} \rangle + \lambda^2 \sum_{k \neq n} \frac{\left| \langle \psi_k^{(0)} | V' | \psi_n^{(0)} \rangle \right|^2}{E_n^{(0)} - E_k^{(0)}} + O(\lambda^3) \tag{2.6}$$

$$|\psi_n(\lambda)\rangle = |\psi_n^{(0)}\rangle + \lambda \sum_{k \neq n} \frac{\langle \psi_k^{(0)} | V' | \psi_n^{(0)} \rangle}{E_n^{(0)} - E_k^{(0)}} |\psi_k^{(0)}\rangle + O(\lambda^2). \tag{2.7}$$

where $E^{(0)}$ is the unperturbed energy and $|\psi^0\rangle$ is a reference state. MBPT is typically computationally less demanding than some other methods and provides a clear hierarchy of corrections, making it valuable for uncertainty quantification. However, the expansion of the exact eigenvalue will break down if one (or more) of the energy denominators become small due to (near-)degeneracies of the unperturbed energies [43].

2.3.2.2 In-Medium Similarity Renormalization Group (IM-SRG)

The IMSRG is a non-perturbative approach that employs a continuous unitary transformation to decouple specific degrees of freedom in the Hamiltonian. The goal is to transform the original Hamiltonian H into a new, effective Hamiltonian $\mathcal{H}(s)$ that is block-diagonal with respect to a chosen reference state (e.g., the ground state or a specific sub-space). This transformation is governed by a flow equation:

$$\frac{d\mathcal{H}(s)}{ds} = [\eta(s), \mathcal{H}(s)] \tag{2.8}$$

where s is a flow parameter and $\eta(s)$ is an anti-Hermitian generator. By evolving the Hamiltonian to a decoupled form, the ground state energy and other observables can be obtained from a simpler, truncated calculation.

2.3.2.3 Coupled Cluster (CC)

Coupled Cluster theory is also a non-perturbative method that also employs a decoupling transformation of the Hamiltonians, however unlike IMSRG, CC uses a non-unitary similarity

transformation. It employs an exponential ansatz for the many-body wavefunction, transforming a reference state $|\Phi_0\rangle$ (e.g., a Hartree-Fock determinant) to the exact ground state $|\Psi\rangle$:

$$|\Psi\rangle = e^T|\Phi_0\rangle \tag{2.9}$$

where T is the cluster operator, $T = T_1 + T_2 + T_3 + \ldots$, with T_k generating k-particle k-hole excitations [12]. The amplitudes defining T are determined by solving a set of coupled non-linear equations derived from the Schrödinger equation projected onto various excited determinants. For example, in Coupled Cluster Singles and Doubles (CCSD), only T_1 and T_2 are included, leading to equations such as:

$$\langle\Phi_0|\bar{H}|\Phi_0\rangle = E_{CC} \tag{2.10}$$

$$\langle\Phi_i^a|\bar{H}|\Phi_0\rangle = 0 \tag{2.11}$$

$$\langle\Phi_{ij}^{ab}|\bar{H}|\Phi_0\rangle = 0 \tag{2.12}$$

where $\bar{H} = e^{-T}He^T$ is the similarity-transformed Hamiltonian.

2.3.2.4 Self-Consistent Green's Function (SCGF)

The Self-Consistent Green's Function (SCGF) approach reformulates the nuclear many-body problem in terms of single-particle propagators. The central object is the one-body Green's function, from which density matrices, energies, and spectral information can be extracted. Its dynamics are governed by the Dyson equation,

$$G(\omega) = G_0(\omega) + G_0(\omega)\,\Sigma(\omega)\,G(\omega), \tag{2.13}$$

where $G_0(\omega)$ is the non-interacting propagator and $\Sigma(\omega)$ the irreducible self-energy encoding correlations. In practice, Σ is computed through diagrammatic expansions such as the Algebraic Diagrammatic Construction (ADC). SCGF provides direct access to spectral functions and neighboring nuclei, and extensions enable applications to open-shell systems and the nuclear continuum. The correlations included at ADC(3) are broadly comparable to IMSRG(2) with perturbative corrections or CCSD(T).

2.3.2.5 Quantum Monte Carlo (QMC)

Quantum Monte Carlo (QMC) methods solve the nuclear many-body problem using coordinate-space wave functions, making them well suited for states with complex intrinsic structure and interactions with high momentum cutoffs. Most QMC calculations first minimize the variational energy (variational principle),

$$min \frac{\langle \psi_T | H | \psi_T \rangle}{\langle \psi_T | \psi_T \rangle} \geq E_0 \tag{2.14}$$

where $|\psi_T\rangle$ is the trial state, then perform imaginary-time projection

$$|\Psi_0\rangle \propto \lim_{\tau \to \infty} e^{-H\tau} |\psi_T\rangle, \tag{2.15}$$

to obtain the ground state quasi-exactly. Different implementations of this projection give rise to methods such as Green's Function Monte Carlo (GFMC) and Auxiliary-Field Diffusion Monte Carlo (AFDMC) [39].

A key challenge is the sign problem: many relevant quantities are not positive definite and cannot be directly sampled as probabilities. This restricts practical QMC to Hamiltonians that are local or contain at most quadratic momentum dependence. Beyond coordinate space, QMC techniques have also been combined with configuration-space approaches, e.g., for sampling MBPT expansions, diagrammatic series, or correlated CC coefficients.

2.3.2.6 Lattice Effective Field Theory (Lattice EFT)

Lattice EFT is a relatively new *ab initio* approach that discretizes spacetime onto a lattice and simulates nucleons and pions directly. This method allows for the direct calculation of nuclear properties from χEFT interactions by employing Quantum Monte Carlo techniques to evaluate the partition function [56]

$$Z(\tau) = \langle \psi | e^{-H\tau} | \psi \rangle, \tag{2.16}$$

where H is the EFT Hamiltonian, typically truncated at a chosen order in the power counting scheme. In practice, Z is expressed as a path integral over sampled field configurations. At large imaginary time τ, ground- and low-lying excited-state information can be extracted, and expectation

values of operators are obtained as [56]

$$\langle O \rangle_\tau = \frac{1}{Z(\tau)} \langle \psi | e^{-H\tau/2} \, O \, e^{-H\tau/2} | \psi \rangle. \tag{2.17}$$

Because of its discretized spatial structure, Lattice EFT is particularly powerful for describing nuclei with complex geometries, such as cluster states [51, 39].

2.4 Essential Role and Challenges of *ab Initio* Methods

Ab initio approaches play a central role in modern nuclear theory because they provide a microscopic and predictive framework for describing nuclear interactions and properties. Starting from nucleon–nucleon and three-nucleon forces derived within chiral effective field theory (χEFT), these methods avoid phenomenological adjustments and instead build on a foundation rooted in QCD. Both the Hamiltonian (through chiral order) and the many-body methods themselves (e.g., CCSD, CCSDT, IMSRG truncations) can be systematically improved, which allows for controlled convergence towards the exact solution. This systematic structure also enables rigorous uncertainty quantification, an essential ingredient for making robust predictions and meaningful comparisons with experiment. Because they are formulated from first principles, *ab initio* methods can reliably predict observables not used in the determination of low-energy constants, including properties of nuclei far from stability. Importantly, they also provide a consistent description of both nuclear structure and nuclear reactions within the same theoretical framework. The ongoing development of more efficient algorithms, the increasing availability of high-performance computing, and advancements in theoretical formalisms are continuously pushing the boundaries of what is achievable with *ab initio* methods, extending their reach to heavier nuclei and more complex nuclear phenomena.

Despite these strengths, *ab initio* methods face persistent challenges. The exponential growth of the Hilbert space limits the tractable system sizes and basis dimensions, leading to enormous computational and memory demands that often require high-performance computing resources. Achieving convergence with respect to basis size and chiral order is computationally expensive, and incomplete convergence introduces uncontrolled uncertainties. The accurate treatment of continuum states and reactions remains more difficult than for bound states, often requiring specialized

extensions of the core formalism. Incorporating higher-body forces, such as four-nucleon interactions, further increases both the complexity of the Hamiltonian and the computational cost. Finally, while uncertainty quantification frameworks exist, refining them to fully capture truncation, approximation, and numerical errors remains an active area of development. These limitations motivate the search for complementary approaches, including modern machine-learning techniques, that can mitigate computational bottlenecks and extend the predictive power of nuclear theory.

CHAPTER 3

QUANTUM COMPUTING

3.1 Introduction

Quantum computing offers a fundamentally new paradigm for information processing based on the principles of superposition, entanglement, and quantum interference. Unlike classical computers, which operate with bits taking values of 0 or 1, quantum computers use qubits that can exist in superpositions of states. This feature, combined with entanglement among multiple qubits, provides an exponential increase in the state space available for computation. These properties make quantum computers especially promising for simulating quantum many-body systems, including nuclei, where the exponential growth of the Hilbert space poses a major challenge for classical methods.

Beyond physics, quantum algorithms have demonstrated transformative potential in other domains—for example, Shor's factoring algorithm [81] showed that quantum computers could efficiently factor large integers, threatening the security of widely used cryptographic schemes such as RSA [6]. Such breakthroughs illustrate the broader computational power of quantum systems while underscoring their potential impact across disciplines.

In the context of nuclear physics, quantum computers have the potential to directly encode and manipulate quantum states, thereby offering new approaches to solving the nuclear many-body problem. Their significance is underscored in the Nuclear Science Advisory Committee's Long Range Plan [67], which highlights quantum computing as a priority area for continued research in nuclear physics. However, present-day devices fall into the so-called noisy intermediate-scale quantum (NISQ) era: they consist of tens to hundreds of qubits but lack full error correction. As a result, they are limited in circuit depth and precision, making error mitigation strategies essential. Looking further ahead, scalable quantum error correction will eventually enable fault-tolerant quantum simulations, but for the near term, progress will rely on hybrid approaches that combine NISQ devices with classical post-processing and mitigation techniques.

3.2 Background on Qubits and Quantum Computation

Broadly speaking, there are two main models of quantum computation: analog and gate-based. Analog approaches, such as quantum annealing, encode problems into an energy landscape and use adiabatic evolution to reach ground states that correspond to solutions of optimization problems. While quantum annealers have demonstrated success for certain combinatorial tasks, they lack the flexibility of a universal computational model. In contrast, gate-based quantum computing generalizes the logic of classical digital circuits by applying unitary operations (quantum gates) to registers of qubits. The gate-based model is universal, meaning that any unitary transformation can be approximated to arbitrary precision with a finite gate set. In this thesis we will focus primarily on gate-based quantum computing, as it provides the most general framework for quantum algorithms relevant to nuclear physics.

At the most fundamental level, a quantum computer is built from quantum bits, or qubits. Unlike classical bits, which can only take the values 0 or 1, a qubit is represented by a normalized vector in a two-dimensional complex Hilbert space [64],

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \qquad \alpha, \beta \in \mathbb{C}, \quad |\alpha|^2 + |\beta|^2 = 1, \tag{3.1}$$

where $|0\rangle$ and $|1\rangle$ form an orthonormal basis. The ability of a qubit to exist in a superposition of states is one of the key features that distinguishes quantum from classical information processing. Geometrically, any pure qubit state can be visualized on the Bloch sphere 3.1, where global phases are neglected and the parameters α and β are mapped onto angular coordinates.

The true computational power of quantum systems emerges when multiple qubits are combined. An n-qubit register is described by the tensor product of n single-qubit Hilbert spaces, giving rise to a state space of dimension 2^n. For example, a two-qubit state has the general form

$$|\psi\rangle = \alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle + \alpha_{10}\,|10\rangle + \alpha_{11}\,|11\rangle\,, \tag{3.2}$$

where the coefficients satisfy $\sum_{i,j} |\alpha_{ij}|^2 = 1$. Importantly, not all multi-qubit states can be written as a product of single-qubit states; such inseparable states are known as entangled. A canonical

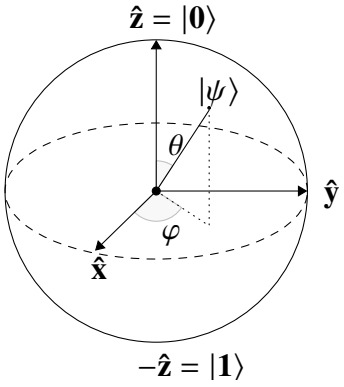

Figure 3.1 Bloch sphere representation of a single qubit.

example is the Bell state

$$\left|\Phi^{+}\right\rangle = \tfrac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right), \tag{3.3}$$

which exhibits correlations that cannot be explained classically. Entanglement is widely regarded as a key resource for quantum computation.

Quantum computations proceed by applying unitary transformations to qubits. The most basic gates include the Pauli operators,

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \tag{3.4}$$

which represent bit-flip, phase-flip, and combined bit- and phase-flip operations, respectively. The Hadamard gate,

$$H = \tfrac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{3.5}$$

creates superpositions, while the two-qubit controlled-NOT (CNOT) gate,

$$\mathrm{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{3.6}$$

acts by flipping the target qubit if the control qubit is in the state $|1\rangle$. Another important single-qubit operation is the T (or $\pi/8$) gate,

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}, \tag{3.7}$$

which introduces a non-Clifford phase crucial for universal quantum computation. Together, gates such as $\{H, T, \text{CNOT}\}$ form a universal set for quantum computation.

As an illustrative example, consider the preparation of the Bell state $|\Phi^+\rangle$. Starting with $|00\rangle$, the first qubit is placed into a superposition by the Hadamard gate,

$$H \otimes I \; |00\rangle = \tfrac{1}{\sqrt{2}} \left(|00\rangle + |10\rangle\right). \tag{3.8}$$

A subsequent CNOT gate, with the first qubit as control and the second as target, yields

$$\text{CNOT}\left(\tfrac{1}{\sqrt{2}} \left(|00\rangle + |10\rangle\right)\right) = \tfrac{1}{\sqrt{2}} \left(|00\rangle + |11\rangle\right) = \left|\Phi^+\right\rangle. \tag{3.9}$$

This simple two-gate circuit 3.2 is one of the most fundamental demonstrations of entanglement.

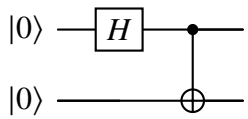

Figure 3.2 Quantum circuit for preparing the Bell state $|\Phi^+\rangle$. The Hadamard gate on the first qubit creates a superposition, and the CNOT gate entangles it with the second qubit.

Finally, measurements extract classical information from a quantum system. Measuring a state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ in the computational basis yields the outcome 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$, as dictated by the Born rule. More generally, measurement collapses the quantum state onto one of the basis states of the chosen observable, destroying coherence in the process.

The combination of superposition, entanglement, and unitary evolution gives quantum computers access to an exponentially large state space: n qubits encode 2^n amplitudes. This scaling rapidly surpasses classical computational resources. For example, storing the full state vector of a 40-qubit system already requires on the order of a terabyte of memory, illustrating why classical

simulation of quantum many-body systems becomes intractable. This exponential advantage motivates the exploration of quantum computing as a tool for nuclear physics, where the Hilbert space of interacting nucleons grows combinatorially with system size.

3.3 Quantum Algorithms

The utility of a quantum computer ultimately rests on the algorithms it can support. A central goal of quantum algorithms in physics is to efficiently extract dynamical or structural information about quantum many-body systems that would be intractable on classical computers. In the nuclear context, these include the computation of ground- and excited-state energies of nuclei, real-time scattering amplitudes, and response functions under external probes. Because of the exponential scaling of Hilbert spaces, classical algorithms quickly become prohibitively expensive, motivating the search for quantum methods that scale more favorably.

One of the earliest and most influential proposals is the quantum phase estimation (QPE) algorithm [70]. QPE allows one to estimate eigenvalues of a unitary operator, which in the case of quantum simulation corresponds to the time-evolution operator $U(t) = e^{-iHt}$. When combined with controlled time evolution, QPE can extract eigenenergies of a Hamiltonian H to exponential precision in the number of qubits used for the phase register. This makes it, in principle, a polynomially efficient method for determining nuclear energy spectra. However, QPE requires long coherent evolutions and fault-tolerant error correction to be practical, placing it firmly in the category of algorithms for future, large-scale quantum computers. More recently, algorithms have been developed with NISQ devices in mind, aiming to efficiently prepare individual eigenstates of general quantum Hamiltonians, such as the Rodeo Algorithm [17].

Closely related to QPE is the task of Hamiltonian simulation: the efficient implementation of $U(t) = e^{-iHt}$. Early approaches used Trotter–Suzuki product formulas [86, 87, 15, 92] to approximate time evolution by sequentially applying exponentials of local Hamiltonian terms. The cost of such algorithms scales polynomially with both simulation time and system size, and more recent advances—such as qubitization and linear-combination-of-unitaries (LCU) methods—have further reduced the complexity [27]. For nuclear physics, Hamiltonian simulation underpins real-

time evolution of nuclear wavefunctions and scattering processes, but like QPE, it requires circuit depths that are beyond the reach of current noisy hardware.

To overcome the limitations of deep circuits in the NISQ era, hybrid quantum-classical algorithms have been introduced that use shallow circuits combined with classical optimization. The most prominent of these is the variational quantum eigensolver (VQE) [69]. VQE employs a parameterized quantum circuit to prepare trial states and then measures expectation values of the Hamiltonian on a quantum device. A classical optimizer updates the parameters to minimize the energy, in direct analogy with the Rayleigh–Ritz variational principle. Because VQE relies on shallow circuits and repeated sampling rather than long coherent evolution, it is considered one of the most practical approaches for near-term nuclear applications, such as computing ground-state energies of light nuclei or exploring effective interactions.

Another promising NISQ-compatible method is quantum imaginary-time evolution (QITE) [62]. QITE seeks to project out ground states by mimicking imaginary-time propagation $e^{-\tau H}$. Since direct non-unitary evolution cannot be implemented on a quantum computer, QITE approximates it through a sequence of unitary operations whose action on local observables matches that of imaginary-time dynamics. This method has already shown success on small-scale systems and offers an attractive alternative to VQE that bypasses some of the challenges associated with noisy variational optimization landscapes.

Beyond these approaches, other quantum algorithms may also find relevance in nuclear applications. The quantum approximate optimization algorithm (QAOA) [16], for example, is designed for combinatorial optimization problems and can be adapted to nuclear structure problems that map naturally onto constrained optimization tasks. Similarly, algorithms for linear systems of equations and quantum machine learning may accelerate specific subroutines in many-body calculations.

Taken together, these methods illustrate two complementary pathways. On the one hand, algorithms like QPE and advanced Hamiltonian simulation offer rigorous polynomial or exponential speedups, but require fault-tolerant devices and thus remain long-term goals. On the other hand, variational and approximate methods such as VQE and QITE are far more hardware-efficient and

align with the capabilities of NISQ-era quantum processors. In nuclear physics, the immediate prospects lie in applying these hybrid approaches to few-body nuclei, benchmarking effective Hamiltonians, and integrating quantum subroutines with classical many-body techniques. In the longer term, the advent of error-corrected quantum computers will make it possible to revisit phase-estimation-based approaches for high-precision nuclear spectroscopy and reaction dynamics.

3.4 Error Correction & Error Mitigation

Quantum computers are fundamentally limited by noise: qubits decohere on microseconds to milliseconds timescales, and gates are implemented with finite precision. These errors accumulate quickly, making long coherent computations infeasible without additional protection. In classical computing, error correction is achieved through redundancy: for example, a single bit can be copied three times and majority-voted to recover from a single error. However, in quantum systems, the no-cloning theorem prohibits the direct copying of an unknown quantum state, preventing straightforward application of classical redundancy [63]. Instead, quantum error correction encodes logical qubits into highly entangled states of multiple physical qubits. Schemes such as the surface code [29] or concatenated codes [47] protect quantum information by distributing it across a larger Hilbert space, enabling errors to be detected and corrected without directly measuring (and thereby destroying) the encoded state. While theoretically robust, these protocols demand significant hardware overhead, often thousands of physical qubits per logical qubit, and are therefore beyond the reach of current noisy intermediate-scale quantum (NISQ) devices.

Given that full error correction is not yet practical, near-term quantum computing relies on error mitigation: strategies that do not eliminate noise but reduce its impact on measured observables. Unlike error correction, mitigation operates without the overhead of large redundancy and is therefore compatible with NISQ hardware. Several methods have been developed and applied in quantum simulation, including zero-noise extrapolation, readout error calibration, probabilistic error cancellation, and task-specific strategies such as Trotter error reduction.

Zero-noise extrapolation (ZNE) is based on the observation that expectation values can often be expressed as smooth functions of the noise rate. Suppose we are interested in an observable

$\langle O \rangle$ measured under a noisy channel $\mathcal{E}_\lambda$, where λ quantifies the effective noise strength. One may artificially amplify the noise by stretching gate durations or repeating gate sequences, yielding data $\langle O \rangle_\lambda, \langle O \rangle_{k\lambda}, \ldots$ that can be fit to a polynomial in λ. Extrapolating back to $\lambda \to 0$ provides an estimate of the ideal expectation value [33]:

$$\langle O \rangle_{\text{ideal}} \approx \sum_{m=0}^{M} c_m \langle O \rangle_{k_m \lambda}, \tag{3.10}$$

where the coefficients c_m are chosen to cancel out noise contributions up to order M. ZNE thus recovers an approximation to the noiseless result without modifying the underlying algorithm.

Readout error calibration addresses the imperfections of qubit measurement, which often misclassify $|0\rangle$ as $|1\rangle$ (and vice versa) with small probability. If we denote the measurement response matrix as

$$M = \begin{pmatrix} 1 - p_{01} & p_{10} \\ p_{01} & 1 - p_{10} \end{pmatrix}, \tag{3.11}$$

then the measured distribution $\vec{q}_{\text{meas}}$ is related to the true distribution $\vec{q}_{\text{true}}$ by $\vec{q}_{\text{meas}} = M\,\vec{q}_{\text{true}}$. By preparing known calibration states (e.g. $|0\rangle$ and $|1\rangle$), one can estimate M and invert it to obtain a corrected estimate of $\vec{q}_{\text{true}}$ [97]. This linear inversion significantly improves the fidelity of probability estimates, particularly for multi-qubit observables.

Probabilistic error cancellation takes a more ambitious approach by attempting to invert the effect of a noisy quantum channel [89]. Any noisy operation $\mathcal{E}$ can be expressed as a linear combination of ideal gates $\mathcal{U}_i$ with quasi-probabilities η_i,

$$\mathcal{E}(\rho) = \sum_i \eta_i \mathcal{U}_i(\rho), \qquad \sum_i |\eta_i| \geq 1. \tag{3.12}$$

By randomly sampling gates $\mathcal{U}_i$ according to a distribution proportional to $|\eta_i|$, one can reproduce the ideal noiseless channel in expectation. The cost of this method is an exponential increase in sampling variance, scaling with the quasi-probability norm $\gamma = \sum_i |\eta_i|$. While demanding in practice, probabilistic error cancellation provides a conceptually exact way to mitigate noise and serves as a benchmark for more efficient approximate techniques.

Finally, for simulation tasks based on Trotterized time evolution, one must contend not only with hardware noise but also with algorithmic error from finite step size. A first-order Trotter decomposition approximates e^{-iHt} for $H = \sum_j H_j$ by

$$e^{-iHt} \approx \left(\prod_j e^{-iH_j \Delta t} \right)^r, \qquad t = r\Delta t, \tag{3.13}$$

introducing a systematic error of order $O(\Delta t^2)$ [74]. Higher-order product formulas or randomized compilation can reduce this error, while polynomial extrapolation in Δt allows one to estimate the limit $\Delta t \to 0$. This idea parallels zero-noise extrapolation but targets algorithmic truncation error rather than hardware imperfections.

These methods can also be used in tandem, since different techniques address distinct sources of noise and can complement one another [57]. In combination, these mitigation strategies provide a practical bridge between the noisy quantum devices of today and the fully error-corrected machines of the future. For nuclear physics applications, they are indispensable: without them, even shallow circuits yield unreliable results. As hardware improves, the continued development of robust mitigation techniques will play a central role in extending the reach of NISQ-era quantum simulations.

3.5 Quantum Machine Learning

Machine learning (ML) and quantum computing share a common challenge: extracting useful patterns from exponentially large state spaces. This overlap has led to increasing interest in quantum machine learning (QML), both as a tool for enhancing quantum devices and as a potential new paradigm for data-driven modeling. In the context of noisy intermediate-scale quantum (NISQ) hardware, ML plays a dual role: it can be used to characterize and mitigate device imperfections, and, conversely, quantum algorithms themselves may provide new approaches to learning from classical or quantum data.

One near-term application is the use of classical ML for error mitigation. Modern quantum devices suffer from highly structured and device-specific noise: error rates vary across qubits, gates, and even temporal conditions. Traditional mitigation strategies such as zero-noise extrapolation

and probabilistic error cancellation assume simplified models of noise, but real hardware often exhibits correlations and non-Markovian dynamics that are difficult to capture analytically. Here, ML methods offer a complementary approach: by training regression or neural-network models on calibration data, one can learn effective noise maps that describe how the device transforms ideal expectation values into measured results. Once trained, such models can be used to denoise observables or guide adaptive circuit compilation strategies [100]. For example, given a set of noisy measurement outcomes $\{x_i\}$, a trained model f_θ can approximate the underlying noiseless distribution,

$$\hat{y} = f_\theta(x), \tag{3.14}$$

where θ are parameters optimized on calibration experiments. This data-driven mitigation bypasses the need for exact physical modeling and can adapt dynamically to device drift.

Beyond supporting current devices, quantum machine learning (QML) itself represents a long-term opportunity for leveraging quantum computers in data analysis. The central idea is that a quantum processor can encode classical or quantum data into high-dimensional Hilbert spaces and manipulate them through unitary transformations, potentially offering exponential feature spaces that are inaccessible to classical kernels. Variational quantum classifiers, quantum Boltzmann machines, and quantum neural networks [61] have been proposed as analogues of classical models, with potential applications ranging from pattern recognition to generative modeling. In physics, QML methods may also enable efficient extraction of physical observables, compression of many-body wavefunctions, and identification of effective Hamiltonians from experimental data.

While much of the promise of QML remains speculative, early demonstrations on small quantum devices show that hybrid quantum-classical models can already capture nontrivial correlations in toy datasets. For nuclear physics, QML could provide powerful tools for both sides of the problem: in the near term, by improving device performance through noise-aware learning, and in the long term, by providing novel frameworks for modeling nuclear structure and dynamics in ways that complement or surpass classical machine learning approaches. In this sense, machine learning and quantum computing are not separate threads but mutually reinforcing technologies that may

accelerate the path toward practical quantum advantage.

CHAPTER 4

MACHINE LEARNING AND OPTIMIZATION

4.1 Introduction

Machine learning (ML) refers to a class of algorithms that extract patterns and build predictive models directly from data. Unlike traditional rule-based programming, ML relies on optimization to learn representations that minimize discrepancies between predictions and observations. In physics, ML has emerged as a powerful tool for tackling high-dimensional, non-linear, and computationally intensive problems. Applications range from accelerating simulations to extracting effective models from large datasets, making it highly relevant for addressing the challenges faced by nuclear many-body methods.

4.2 Optimization

At the core of ML lies optimization: the process of adjusting parameters to minimize a loss function. Broadly speaking, optimization methods can be classified along two axes. The first is between *local* and *global* strategies [93]. Local optimizers refine parameters within the vicinity of an initial guess and are therefore efficient but prone to becoming trapped in suboptimal minima when the loss landscape is rugged. Global optimizers, by contrast, are designed to explore the parameter space more broadly and reduce the likelihood of stagnation, often at the expense of slower convergence. The second axis is between *gradient-based* and *gradient-free* approaches [93]. Gradient-based methods explicitly use derivatives of the loss function, whereas gradient-free methods operate without them and are suitable when derivatives are unavailable, noisy, or prohibitively expensive to compute.

Machine learning methods are commonly categorized into supervised, unsupervised, and reinforcement learning. In supervised learning, models are trained on labeled data to learn a mapping between inputs and outputs. Unsupervised learning instead discovers patterns or structure in unlabeled data, such as clustering or dimensionality reduction, while reinforcement learning trains an agent to make sequential decisions through trial and error based on reward feedback. There are other learning paradigms as well, but we restrict our discussion to supervised learning in this

work [60].

Supervised learning, can be posed as an optimization problem. Given parameters $\theta \in \mathbb{R}^d$ and a differentiable loss $\mathcal{L} : \mathbb{R}^d \to \mathbb{R}$, the goal is

$$\min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta). \tag{4.1}$$

A point $\theta^\star$ is a (global) minimizer if $\mathcal{L}(\theta^\star) \leq \mathcal{L}(\theta)$ for all θ [60]; it is a local minimizer if this inequality holds in some neighborhood of $\theta^\star$. For smooth $\mathcal{L}$, Fermat's theorem states that a necessary condition for optimality is $\nabla \mathcal{L}(\theta^\star) = 0$ [60], and when $\nabla^2 \mathcal{L}(\theta^\star) \succeq 0$ the point is at least a local minimum. If $\mathcal{L}$ is convex, any stationary point is global; if $\mathcal{L}$ is strictly (or strongly) convex, the minimizer is unique.

4.2.1 Convex Least Squares: Problem and Closed-Form Solution

Consider ordinary least squares (OLS). With data $\{(x_i, y_i)\}_{i=1}^N$ where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, stack rows into the design matrix $X \in \mathbb{R}^{N \times d}$ and targets into $y \in \mathbb{R}^N$. The linear model $\hat{y} = X\theta$ yields the mean squared error

$$\mathcal{L}(\theta) = \frac{1}{N} \|y - X\theta\|_2^2. \tag{4.2}$$

This objective is a convex quadratic in θ with gradient and Hessian

$$\nabla \mathcal{L}(\theta) = -\frac{2}{N} X^\top (y - X\theta), \qquad \nabla^2 \mathcal{L}(\theta) = \frac{2}{N} X^\top X \geq 0. \tag{4.3}$$

Setting the gradient to zero gives the normal equations

$$X^\top X \, \theta^\star = X^\top y. \tag{4.4}$$

If $X^\top X$ is invertible (e.g., X has full column rank), the unique convex least squares solution is [60]

$$\theta^\star = (X^\top X)^{-1} X^\top y. \tag{4.5}$$

When $X^\top X$ is singular, the minimum-norm solution is given by the Moore–Penrose pseudoinverse X^+:

$$\theta^\star = X^+ y, \qquad X^+ \equiv (X^\top X)^+ X^\top. \tag{4.6}$$

While ordinary least squares (OLS) provides a simple and analytically tractable approach to supervised learning, it does not necessarily guarantee good generalization to unseen data. To address issues such as overfitting, bias–variance tradeoffs, and nonlinear relationships, more sophisticated methods have been developed, including kernel ridge regression, support vector regression (SVR), Lasso, and other regularization-based techniques. Nevertheless, we examine OLS here for its mathematical simplicity, which allows for clear analytical demonstrations and theoretical insight.

4.2.2 First-Order (Gradient-Based) Solution Path

Gradient-based methods update parameters using the derivative of the loss. The simplest instance is gradient descent:

$$\theta_{t+1} = \theta_t - \eta \nabla \mathcal{L}(\theta_t). \tag{4.7}$$

Applied to OLS and its gradient(4.3), the update becomes

$$\begin{aligned} \theta_{t+1} &= \theta_t - \eta \left(-\tfrac{2}{N} X^\top (y - X\theta_t) \right) \\ &= \theta_t + \tfrac{2\eta}{N} X^\top (y - X\theta_t). \end{aligned} \tag{4.8}$$

which is summarized in Algorithm 4.1, for a linear regression example. To show that gradient decent converges to $\theta^\star$, first we define the error $e_t = \theta_t - \theta^\star$. Using the normal equations $X^\top (y - X\theta^\star) = 0$, one can show

$$e_{t+1} = \theta_{t+1} - \theta^\star \tag{4.9}$$

$$= \theta_t - \theta^\star + \tfrac{2\eta}{N} X^\top \big(y - X\theta_t\big) \tag{4.10}$$

$$= e_t + \tfrac{2\eta}{N} X^\top \big(y - X(e_t + \theta^\star)\big) \tag{4.11}$$

$$= e_t - \tfrac{2\eta}{N} \big[X^\top y - X^\top X e_t - X^\top X\theta^\star\big] \tag{4.12}$$

$$= e_t - \tfrac{2\eta}{N} \big[X^\top y - X^\top X e_t - X^\top y\big] \tag{4.13}$$

$$= e_t - \tfrac{2\eta}{N} X^\top X e_t \tag{4.14}$$

$$= \Big(I - \tfrac{2\eta}{N} X^\top X\Big) e_t. \tag{4.15}$$

Thus the error contracts by multiplication with a fixed linear operator. Convergence is therefore governed by the spectral radius of $I - \frac{2\eta}{N} X^\top X$ [8, 65]. Specifically, if

$$0 < \eta < \frac{1}{\lambda_{\max}\left(\frac{2}{N} X^\top X\right)}, \tag{4.16}$$

then all eigenvalues of the update matrix lie in $(-1, 1)$, and the error converges linearly to zero:

$$\|e_t\| \leq \rho^t \|e_0\|, \qquad \rho = \max_i \left|1 - \tfrac{2\eta}{N} \lambda_i(X^\top X)\right| < 1. \tag{4.17}$$

This update can be written compactly as $e_{t+1} = A e_t$, where $A = I - \frac{2\eta}{N} X^\top X$. The resulting recursion is a linear iteration, sharing the same mathematical form as the power method [91]. However, unlike the power iteration, which amplifies the dominant eigenvector, gradient descent operates with eigenvalues strictly inside the unit circle, contracting all components of the error. Thus, its convergence rate is governed by the spectral radius of A, ensuring linear convergence to the unique least-squares solution $\theta^\star$.

First-order methods are simple, cheap per iteration, and easy to implement, but the rate of convergence depends on the condition number of $X^\top X$. When $X^\top X$ is ill-conditioned, gradient descent may require many iterations. To accelerate convergence, one often augments gradient descent with *momentum*, which effectively damps oscillations and exploits curvature information implicitly. Classical schemes include Polyak's heavy-ball method and Nesterov's accelerated gradient, both of which improve the dependence on the condition number [59]. In large-scale machine learning, adaptive methods such as AdaGrad, RMSProp, and Adam further tune learning rates based on past gradient magnitudes, making them robust to non-stationarity and ill-conditioning [37]. Although these methods do not reach the one-step optimality of Newton's method, they often achieve a favorable balance between per-iteration cost and convergence speed in practice.

Algorithm 4.1 Gradient Descent for Linear Regression

Input: data matrix $X \in \mathbb{R}^{N \times d}$, targets $y \in \mathbb{R}^N$, learning rate η, iterations T
Initialize parameters $\theta_0 \in \mathbb{R}^d$
for $t = 0, 1, 2, \ldots, T-1$ **do**
  Compute residual $r_t = y - X\theta_t$
  Compute gradient $g_t = -\frac{2}{N} X^\top r_t$
  Update $\theta_{t+1} = \theta_t - \eta g_t$
end for
Output: θ_T

4.2.3 Second-Order (Newton-Type) Solution Path

Second-order methods incorporate curvature information through the Hessian $H(\theta) = \nabla^2 \mathcal{L}(\theta)$. Newton's method performs the iteration

$$\theta_{t+1} = \theta_t - H(\theta_t)^{-1} \nabla \mathcal{L}(\theta_t). \tag{4.18}$$

For OLS, the Hessian is constant (4.3):

$$H(\theta) = \tfrac{2}{N} X^\top X. \tag{4.19}$$

Substituting both gradient and Hessian, the Newton update becomes

$$\begin{aligned} \theta_{t+1} &= \theta_t - \left(\tfrac{2}{N} X^\top X\right)^{-1} \left(-\tfrac{2}{N} X^\top (y - X\theta_t)\right) \\ &= \theta_t + (X^\top X)^{-1} X^\top (y - X\theta_t) \\ &= \theta_t + (X^\top X)^{-1} X^\top y - (X^\top X)^{-1} X^\top X \theta_t \\ &= (X^\top X)^{-1} X^\top y \\ \theta_{t+1} &= \theta^\star \end{aligned} \tag{4.20}$$

Thus, for any starting point θ_t: Newton's method converges to the OLS solution in a *single iteration* (up to numerical error in solving the linear system).

The one-step convergence hinges on explicitly forming and inverting (or factorizing) the Gram matrix $X^\top X$. This operation costs $O(d^3)$, where d is the number of features, which is prohibitive in very high dimensions (4.1), though efficient factorizations (e.g., Cholesky, QR) are widely used in practice [91]. For large-scale problems, quasi-Newton methods (e.g., BFGS, L-BFGS [99])

approximate H^{-1} iteratively, achieving faster convergence than gradient descent while avoiding the full matrix inversion.

Second-order methods exploit curvature to achieve dramatically faster convergence. For quadratic objectives such as OLS, Newton's method is optimal in the sense of recovering the exact minimizer in a single step. The trade-off is computational: explicitly handling the Hessian becomes impractical in high dimensions, motivating approximations that balance speed and scalability. From a broader perspective, momentum-based first-order methods (e.g., Adam) can be viewed as lightweight, scalable alternatives that capture some curvature information without ever forming the Hessian, thereby bridging the gap between basic gradient descent and full Newton-type schemes [59, 37].

4.2.4 Local vs. Global; Gradient-Free Methods

A *local optimizer* refines parameters in the neighborhood of an initialization and may converge to a nearby local minimum in nonconvex problems. A *global optimizer* employs exploration mechanisms to reduce sensitivity to initialization and to escape local minima, typically at higher computational cost. Methods are *gradient-based* when they use derivatives (e.g., gradient descent, Newton, quasi-Newton) and *gradient-free* when they do not (e.g., Nelder–Mead and Powell as local direct-search methods; simulated annealing and basin-hopping as global stochastic strategies) [93]. In convex quadratic problems like OLS, any local method that finds a stationary point attains the global minimizer $\theta^\star$.

When gradients are not available or reliable, one may resort to direct search techniques. Nelder–Mead, for instance, evolves a simplex in parameter space and is effective in problems of moderate dimensionality. Powell's method iteratively refines a set of search directions to minimize along coordinate lines, again without requiring derivatives. Both are local in nature and their convergence depends strongly on initialization.

Global optimization methods aim to escape local minima by introducing stochasticity or randomized exploration. Simulated annealing explores the loss surface with a temperature parameter that gradually decreases, allowing the algorithm to accept uphill moves early on and thus avoid

premature convergence. Basin-hopping alternates between random perturbations and local minimization steps, effectively searching for the lowest-energy basin in a rugged landscape. These approaches are considerably more computationally intensive but are well-suited to highly non-convex problems where local methods are insufficient.

4.3 Machine Learning Models

A wide range of ML models have been developed, from traditional approaches such as decision trees, support vector machines, and Gaussian processes to more flexible neural networks (NNs) [32, 60, 73]. NNs consist of layers of interconnected nodes with non-linear activation functions. Each layer transforms its input through an affine transformation followed by a non-linear map, enabling the network to approximate highly non-linear functions. The depth and width of a network determine its expressive power, and under mild assumptions, even shallow NNs are universal function approximators. In practice, deep networks are trained via stochastic gradient descent and its variants, using backpropagation to compute gradients of a chosen loss function with respect to millions of parameters. Regularization techniques, such as dropout or weight decay, mitigate overfitting, while architectural innovations, convolutional layers, recurrent units, attention mechanisms, tailor NNs to structured data such as images, sequences, and graphs. Despite their success, NNs face challenges: training is non-convex and often sensitive to hyperparameters, while interpretability remains limited compared to more transparent statistical models [34, 58].

4.3.1 Neural Networks

A neural network (NN) is a parametric function approximator composed of layers of affine transformations and non-linear activation functions. Given an input vector $x \in \mathbb{R}^{d_{\text{in}}}$, a feedforward neural network with L layers produces an output $f(x;\theta)$ through successive transformations

$$h^{(0)} = x, \tag{4.21}$$

$$h^{(\ell)} = \sigma\left(W^{(\ell)} h^{(\ell-1)} + b^{(\ell)}\right), \quad \ell = 1, \ldots, L-1, \tag{4.22}$$

$$f(x;\theta) = W^{(L)} h^{(L-1)} + b^{(L)}. \tag{4.23}$$

Here $W^{(\ell)}$ and $b^{(\ell)}$ are the weights and biases of layer ℓ, σ is a non-linear activation function (e.g. ReLU, tanh, sigmoid Figure 4.1), and $\theta = \{W^{(\ell)}, b^{(\ell)}\}_{\ell=1}^{L}$ denotes all trainable parameters.

xxyyy

Figure 4.1 Left: ReLU activation function $\mathrm{ReLU}(x) = \max(0, x)$. Middle: Sigmoid activation function $\sigma(x) = \frac{1}{1+e^{-x}}$. Right: Hyperbolic tangent activation $\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$.

The introduction of non-linearity via σ is crucial: without it, the network reduces to a single affine transformation, regardless of depth. With non-linear activations, however, neural networks are universal approximators: even a single hidden layer with enough neurons can approximate any continuous function on a compact set to arbitrary accuracy.

Training a neural network amounts to solving the optimization problem

$$\min_{\theta} \ \mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell\big(f(x_i; \theta), y_i\big), \tag{4.24}$$

where $\ell(\cdot, \cdot)$ is a loss function measuring the discrepancy between prediction $f(x_i; \theta)$ and target y_i. Gradients $\nabla_\theta \mathcal{L}$ are computed efficiently using *backpropagation*, an application of the chain rule that propagates sensitivities from the output layer back through the network [76]. Parameters are updated using (stochastic) gradient descent or its variants.

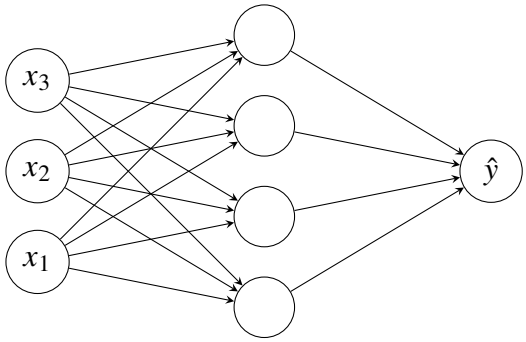

Figure 4.2 A simple feedforward neural network with one hidden layer.

Neural networks are highly expressive but introduce a rugged, non-convex optimization landscape. While gradient descent variants are widely effective, convergence and generalization often depend on careful tuning of hyperparameters (learning rate, architecture, regularization). Never-

theless, the flexibility of NNs has made them the dominant model in modern machine learning, especially in high-dimensional domains such as images, sequences, and physical simulations.

4.4 Machine Learning in Physics

ML methods have been increasingly applied in physics to complement or accelerate traditional approaches. Gaussian processes provide a probabilistic framework for regression and uncertainty quantification, making them attractive for tasks such as extrapolating nuclear observables. Neural networks, including physics-informed neural networks (PINNs), incorporate conservation laws or symmetries into the architecture or loss function, ensuring physically consistent predictions [11, 10, 55]. Neural quantum states (NQS) represent another frontier, where NNs serve as variational ansätze for many-body wave functions, bridging modern ML with quantum many-body theory [49, 44, 53].

An important application of ML in physics is the construction of emulators, or surrogate models, that replicate the behavior of computationally intensive simulations at greatly reduced cost. In nuclear physics, such emulators can approximate ab initio calculations or effective field theory (EFT) predictions, providing rapid evaluations necessary for Bayesian inference. This is particularly critical when extracting constraints on low-energy constants (LECs), where each evaluation of the likelihood may require solving the Schrödinger equation or running a many-body calculation. By training an emulator on a representative set of simulation outputs, one obtains a fast surrogate that can be queried millions of times within a Bayesian sampling routine.

Emulators also play a central role in uncertainty quantification [79, 78, 66, 82, 2]. In Bayesian inference, the total uncertainty arises from multiple sources: experimental error in the measured data, model error stemming from approximations in the theoretical framework, and emulator error associated with the surrogate approximation itself. A careful treatment of these uncertainties is necessary to provide credible intervals on extracted parameters, such as LECs, and to assess the predictive power of the underlying theory. Recent work has demonstrated how Gaussian process emulators or neural network surrogates can be embedded directly into Bayesian pipelines, enabling systematic propagation of uncertainties while reducing the prohibitive cost of repeated full-scale simulations.

4.4.1 Outlook

While existing ML approaches have demonstrated impressive results, their application to physics faces limitations. Many models are purely data-driven and may fail to respect underlying symmetries or conservation laws. Physics-informed neural networks (PINNs), while promising, are often over-parameterized and computationally costly, sometimes slower and less accurate than simply reducing the fidelity of the underlying high-fidelity solver. Gaussian processes (GPs), though mathematically elegant, suffer from unfavorable computational scaling: training requires inverting an $N \times N$ covariance matrix, leading to $O(N^3)$ cost with the number of training points. Moreover, GP performance degrades in the presence of near-degenerate training data, where interpolation becomes ill-conditioned, and their extrapolation ability outside the training domain is generally poor. More broadly, optimization in high-dimensional parameter spaces can lead to overfitting and weak generalization. These challenges underscore that ML methods cannot be applied as black boxes: they must be carefully tailored to the structure and demands of physical systems. The goal of this thesis is to advance such tailored approaches, combining physical insight with the flexibility of modern learning methods, and designed to overcome shortcomings of current *ab initio* techniques while remaining consistent with the principles of nuclear many-body theory.

CHAPTER 5

TENSOR NETWORKS

5.1 Introduction

In addition to machine learning and quantum computing, another framework has been developed within quantum mechanics to address many-body problems: tensor networks (TNs) [9, 4]. Originally motivated by the need to efficiently represent entangled quantum states, tensor networks have since grown into a unifying language across physics, computation, and data science. They were first introduced to reduce the exponential complexity of Hilbert space, but their mathematical structure has proved useful well beyond quantum mechanics, with applications now emerging in machine learning and the solution of partial differential equations. TNs also presents a straightforward and transparent notation used to describe them that utilizes a graphical language, where the structure is manifest. Many general properties of the objects under study, particularly quantum states, can be identified directly from the structure of the network needed to describe them.

The core motivation arises from the *curse of dimensionality* that plagues quantum many-body physics, that arise from the exponential growth of Hilbert space with system size. A system of N qubits requires 2^N complex amplitudes to fully specify, which quickly becomes intractable. Tensor networks exploit the fact that states of physical interest, such as ground states of local Hamiltonians, tend to have limited entanglement. By factorizing high-order tensors into networks of low-rank tensors, TNs retain the essential entanglement structure while discarding redundant degrees of freedom.

The underlying principle is closely related to the singular value decomposition (SVD). When a bipartition of a system admits only a small number of significant Schmidt coefficients, the state can be approximated with a low-rank factorization. Tensor networks generalize this idea to many bipartitions simultaneously, providing scalable approximations that are both physically and computationally meaningful. Over the past three decades, they have become central to condensed matter physics and quantum information, and even machine learning. Forming the foundation of algorithms such as the density matrix renormalization group (DMRG) and the time-evolving block

decimation (TEBD).

5.2 Matrix Product States (MPS)

Tensors are a generalisation of vectors and matrices. A d-dimensional vector $V \in \mathbb{C}^d$, and a $n \times m$-dimensional matrix $M \in \mathbb{C}^{n\times m}$. Correspondingly a rank-r tensor $T \in \mathbb{C}^{d1\times\cdots\times dr}$. We can clearly see that scalars, vectors and matrices are all therefore rank 0, 1 and 2 tensors respectively.

The most basic and widely used tensor network is the *matrix product state* (MPS), also known as the *tensor train*. Consider a chain of N quantum spins. A generic wavefunction can be expanded as

$$|\psi\rangle = \sum_{i_1,\ldots,i_N=0}^{1} C_{i_1 i_2 \ldots i_N} \, |i_1 i_2 \ldots i_N\rangle, \tag{5.1}$$

where the coefficient tensor C has 2^N entries. An MPS factorizes this high-rank tensor into a product of low-rank tensors [9, 4]:

$$C_{i_1 i_2 \ldots i_N} = \sum_{\{\alpha\}} A^{[1]}_{i_1,\alpha_1} A^{[2]}_{i_2,\alpha_1,\alpha_2} \cdots A^{[N]}_{i_N,\alpha_{N-1}}. \tag{5.2}$$

Here each $A^{[k]}$ is a three-index tensor (except the boundaries, which have two indices). The internal indices α_j of dimension χ are called *bond dimensions*, controlling how much entanglement the MPS can capture. When $\chi = 1$, the state is a simple product state; larger χ allow for more entangled states.

Entanglement and SVD

The efficiency of MPS is explained by entanglement entropy. For a bipartition $A|B$, the Schmidt decomposition reads

$$|\psi\rangle = \sum_{\alpha=1}^{\chi} \lambda_\alpha |\phi^A_\alpha\rangle \otimes |\phi^B_\alpha\rangle, \tag{5.3}$$

with Schmidt coefficients (weights) λ_α. The entanglement entropy is

$$S_A = -\sum_\alpha \lambda_\alpha^2 \log \lambda_\alpha^2. \tag{5.4}$$

If S_A is small, only a few λ_α are significant, allowing $|\psi\rangle$ to be well-approximated with a small bond dimension χ. Note that the Schmidt coefficients λ corresponds precisely to the singular value of the decomposition of (5.3).

One can perform successive singular value decompositions along each bipartition cut in turn, splitting out the tensor into local tensors $A^{[i]}$, and diagonal matrices of singular values λ quantifying the entanglement across that cut. This can be reapeated until we obtain Figure 5.1, which then can be contracted efficently back to the generic form $|\psi\rangle$. This is what is known as an MPS.

One of the most powerful results in the field of tensor network analytics is a complete classification of gapped phases in 1D. For the result above This is precisely follows what happens for gapped 1D systems, where entanglement entropy obeys an *area law*. Thus, MPS are not generic representations of all states, but are particularly efficient for physically relevant low-entanglement states. In higher dimensions, projected entangled pair states (PEPS) generalize the MPS construction, though contraction costs grow substantially and scale with volume.

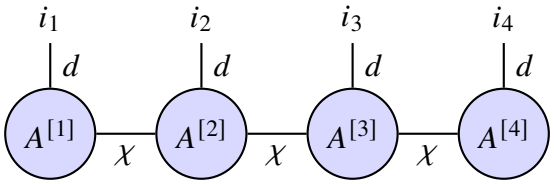

Figure 5.1 Matrix product state (MPS). Each site tensor $A^{[k]}$ has one physical leg of dimension d and two virtual legs of bond dimension χ.

5.3 Beyond MPS: DMRG and TEBD

5.3.1 Density Matrix Renormalization Group (DMRG)

Building on the MPS framework, a number of powerful algorithms have been developed. The density matrix renormalization group (DMRG), introduced by White in the 1990s [94, 95], is essentially a variational optimization scheme within the MPS class. It is grounded in the variational principle (2.14), seeking the MPS that minimizes the expectation value of the Hamiltonian for a fixed bond dimension. It iteratively updates local tensors to minimize the ground-state energy of a Hamiltonian and has become the benchmark method for studying strongly correlated one-dimensional systems [9].

In quantum many-body physics, DMRG has become a cornerstone for obtaining high-accuracy results in low-dimensional systems. In quantum chemistry, it is employed to solve the electronic

Schrödinger equation, allowing systematic convergence toward full configuration interaction (FCI) accuracy in large active spaces [13, 14]. Beyond condensed matter and chemistry, DMRG and its tensor network generalizations are increasingly used in *ab initio* nuclear structure calculations, where they can serve as highly accurate solvers for Hamiltonians derived from other many-body frameworks [40, 39].

DMRG's strength lies in its ability to efficiently represent low-entanglement states by systematically truncating the Hilbert space according to the spectrum of the reduced density matrix. This makes it particularly effective for gapped 1D quantum lattice models, such as the Heisenberg or Hubbard chains, where it achieves near-exact ground states with modest computational resources. Over time, extensions of DMRG to finite temperature, time evolution, and two-dimensional geometries have broadened its scope across condensed matter and quantum chemistry.

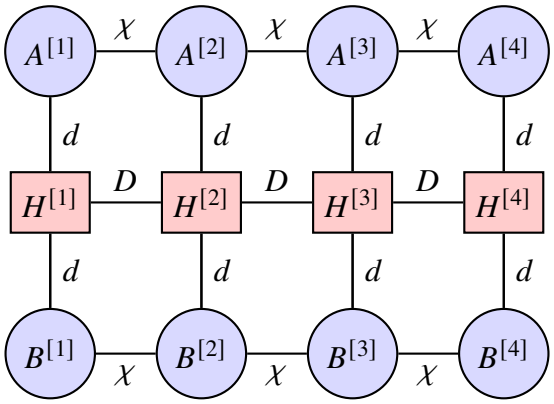

Figure 5.2 DMRG sandwich: bra MPS (top) and ket MPS (bottom) are contracted with the Hamiltonian represented as an MPO (middle chain of red squares). Horizontal bonds carry dimension χ for the MPS and D for the MPO.

5.3.2 Time-Evolving Block Decimation (TEBD)

The time-evolving block decimation (TEBD) algorithm [**?**, **?**] extends the MPS framework from static to dynamical simulations. While DMRG finds ground states variationally, TEBD focuses on the real- or imaginary-time evolution of quantum states. It achieves this by decomposing the time-evolution operator e^{-iHt} using a Trotter-Suzuki expansion into a sequence of local two-site unitaries that can be efficiently applied to an MPS. After each local update, a singular value

decomposition (SVD) is performed to truncate small Schmidt coefficients and control the bond dimension, ensuring computational tractability while maintaining accuracy.

TEBD has proven particularly effective for simulating non-equilibrium dynamics in one-dimensional lattice systems, quantum quenches, and transport phenomena. In imaginary time, it can also be used to obtain ground states, making it closely related to DMRG in spirit. Modern variants such as the time-dependent variational principle (TDVP), an alternative to the Lie-Trotter decomposition, provide more stable and geometrically consistent formulations of MPS dynamics by directly integrating the Schrödinger equation projected onto the MPS manifold [35, 36].

Beyond its role in fundamental research, TEBD and related tensor-network time-evolution algorithms are increasingly used to benchmark quantum hardware. By classically simulating shallow-depth quantum circuits, they help delineate the frontier between classically tractable and quantum-advantage regimes, providing context for claims of quantum supremacy by platforms such as Google and IBM [3, 45, 68, 90].

5.4 Tensor Networks in Machine Learning and Quantum Computing

Tensor networks are not confined to quantum many-body physics. Their mathematical structure makes them naturally suited for problems in machine learning and quantum computing, where high-dimensional data and complex entanglement-like correlations must be represented efficiently.

In machine learning, the relationship between tensor networks and neural networks has become increasingly clear. Neural network layers can often be reformulated as tensor contractions, and weight matrices can be decomposed into tensor network forms. This reduces the number of parameters while preserving expressive power, providing a principled form of model compression. Architectures such as tree tensor networks (TTN) and multiscale entanglement renormalization ansatz (MERA) mirror the hierarchical feature extraction of deep learning models, and have been explored as interpretable alternatives to conventional networks [85, 52, 80].

A critical aspect of applying tensor networks to both physics and machine learning is the problem of tensor contraction paths. Contracting a tensor network corresponds to summing over internal indices, but the order in which contractions are performed can drastically affect compu-

tational cost. Finding an optimal contraction path is generally an NP-hard problem, yet heuristic and algorithmic strategies have been developed that make contraction feasible in practice [72]. Recent advances in contraction path optimization have made it possible to simulate larger quantum circuits classically and to scale tensor-network-inspired machine learning models. In the context of deep learning, optimized contraction paths allow large-scale tensorized models to be trained and evaluated efficiently, making them competitive with standard architectures.

In quantum computing, tensor networks also play a dual role. On the one hand, they provide efficient classical simulation tools for low-entanglement quantum circuits, enabling benchmarking of near-term devices. On the other hand, they inspire quantum variational ansätze, such as MPS-like or PEPS-like structures embedded directly into parameterized quantum circuits. These hybrid approaches exploit the efficiency of tensor network representations while leveraging the flexibility of quantum hardware.

For the purposes of this thesis, tensor networks will also serve as a bridge to parametric matrix models (PMMs). Large parameter spaces can be compressed through tensor decompositions, and higher-order tensors that appear in surrogate governing equations can be managed using network architectures with efficient contraction paths. In this way, tensor networks are not only a tool for quantum many-body physics but also a methodological link connecting machine learning, quantum computation, and the PMM framework.

CHAPTER 6

PARAMETRIC MATRIX MODELS

6.1 Introduction

Nuclear physics presents a suite of challenging computational problems. One frequently encounters high-dimensional matrices with sparse structure, making decomposition and diagonalization both computationally demanding and numerically delicate. These issues are amplified in ab initio approaches, where accurate modeling of many-body interactions is essential. In particular, the exponential growth of Hilbert-space dimensions, the sign problem in Monte Carlo lattice simulations, and the need to solve large-scale sparse eigenvalue systems exemplify the complexity of the nuclear many-body problem. To address these challenges, and motivated by the transformative potential of artificial intelligence (AI) and machine learning (ML) in this domain, we have developed a machine learning framework that aligns more naturally with the unique computational and physical structure of nuclear systems.

In this thesis, the primary focus is on developing parametric matrix models that can efficiently and robustly address the computational complexities of nuclear many-body physics [23]. By leveraging machine learning techniques tailored to the problem's inherent structure, this work aims to overcome traditional computational barriers and enable high-precision simulations in nuclear theory. This effort reflects a broader shift toward integrating AI and ML methodologies into nuclear physics to enhance both predictive accuracy and computational efficiency.

A key aspect of this integration is the development of fast emulators, which are essential for uncertainty quantification tasks such as Bayesian analysis. These emulators can approximate complex models with high fidelity, enabling rapid parameter-space exploration and providing reliable statistical estimates of uncertainty. Together, these advances promote faster, more accurate, and more interpretable simulations, paving the way for next-generation tools in many-body nuclear theory.

The remainder of this chapter is organized into two main parts. First, we present the theoretical foundations of Parametric Matrix Models (PMMs), outlining their core principles and methodolog-

ical framework. Next, we demonstrate the versatility of PMMs by applying them to a diverse set of problems. These applications range from addressing the eigenvalue problem in many-body systems and simulating Monte Carlo lattice Euclidean time evolution, to mitigating errors in quantum computing, analyzing the computational fluid dynamics of Navier-Stokes equations, and tackling general machine learning tasks, including regression and classification. This broad exploration highlights the adaptability of PMMs as a powerful tool across various computational challenges.

6.2 Theory

A fundamental step in addressing any physics problem is identifying the governing equations. Although these equations often yield complex phenomena in their solutions, their intrinsic structure is typically simple and logical. This structure imposes significant constraints on the analytic behavior of the solutions, such as enforcing symmetries, conserved quantities, causality, and analyticity. Unfortunately, standard machine learning algorithms, who model the solution, frequently fail to reproduce these critical constraints, leading to inefficiencies and reduced accuracy in scientific computing tasks.

To overcome these challenges, various physics-informed machine learning techniques have been developed. For instance, equivariant neural networks are designed to embed certain constraints into their architecture. Yet, when it comes to modeling more complicated physical laws, it becomes increasingly difficult to construct neural networks that inherently adhere to all necessary principles. Consequently, many approaches, physics informed neural network (PINNs), rely on penalizing deviations from the expected behavior, an approach that cannot guarantee exact compliance. Moreover, a common shortfall of many deep learning methods is their lack of interpretability, a significant drawback when precision and insight are vital.

In this work, we introduce a novel class of machine learning algorithms termed *parametric matrix models* (PMMs). PMMs address these limitations by replacing the operators within known or hypothesized governing equations with trainable, parameterized counterparts. By leveraging the principles of model order reduction and reduced basis methods, PMMs construct efficient approximate matrix equations in finite dimensions. In theory, methods like the proper orthogonal

decomposition can be used to guarantee the existence and structure of these reduced equations.

Unlike most machine learning approaches that optimize explicit functional forms for outputs, PMMs belong to a class of implicit function-based models. A PMM is defined as a set of matrix equations with unknown parameters, which are optimized to encapsulate the desired outputs as implicit functions of the input features. When the form of the governing equations is known, the structure of the PMM can be designed to mirror these equations, thereby enhancing both interpretability and physical consistency.

Parametric matrix models can incorporate extensive mathematical and scientific priors from the outset, or be deployed more generally as efficient universal function approximators. Although matrix-based methods have long been employed for dimensionality reduction in machine learning, PMMs represent a fresh perspective by grounding the approach in fundamental physical principles and leveraging the implicit functions derived from matrix equations.

The theoretical foundation of the Parametric Matrix Model (PMM) lies within the broader class of Reduced Basis Methods (RBMs). We adopt the term implicit RBMs following the interpretation by Patrick Cook [24, 22]. To derive the governing equations of the PMM, one introduces a linear projection operator that maps the high-fidelity solution space onto a lower-dimensional latent space that defines the PMM. The basis for this projection can be constructed using various decomposition techniques, such as Proper Orthogonal Decomposition (POD), Nonnegative Matrix Factorization (NMF), or others, depending on the problem structure and desired properties.

For illustration, and throughout the remainder of this thesis, we adopt the POD basis. To demonstrate the underlying idea, consider a simple linear differential initial value problem given by

$$\frac{du}{dt} = A(\vec{x})u + b(\vec{x}), \qquad u(0) = u_0 \tag{6.1}$$

where $u \in \mathbb{C}^N$, $A(\vec{x}) \in \mathbb{C}^{N\times N}$, $b(\vec{x}) \in \mathbb{C}^N$, and $\vec{x} \in \mathbb{R}^p$. This formulation is deliberately kept general so both the matrix A and vector b are complex-valued functions, each dependent on the real model parameters encapsulated in $\vec{x}$. The POD framework begins by collecting M snapshots of the solution to the differential equation 6.1. These snapshots may represent solution vectors taken at

different times, for different parameter values, or both. For simplicity, we assume the snapshots are collected over time for a fixed $\vec{x}$. forming the data matrix

$$\mathrm{X} = \begin{bmatrix} | & | & & | \\ u(t_1) & u(t_2) & \cdots & u(t_M) \\ | & | & & | \end{bmatrix} \tag{6.2}$$

where $\mathrm{X} \in \mathbb{C}^{N \times M}$. The POD technique projects the original high-dimensional problem onto a lower-dimensional subspace spanned by the first n principle components principal components (i.e., the n left singular vectors) of this data matrix. Using the singular value decomposition (SVD),

$$\mathrm{X} \overset{\mathrm{SVD}}{\rightarrow} U\Sigma V^{\dagger} \tag{6.3}$$

we define the POD projector as

$$P = U[:, 1:n] \tag{6.4}$$

which consists of the first n columns of U (assuming the singular values are arranged in descending order). By projecting our operators and initial conditions, we obtain the reduced operators $\hat{A}(\vec{x}) = P^{\dagger}A(\vec{X})P$, $\hat{b} = P^{\dagger}b(\vec{x})$, and the initial state $\hat{u}_0 = P^{\dagger}u_0$. The reduced order model (ROM) becomes

$$\frac{d\hat{u}}{dt} = \hat{A}(\vec{x})\hat{u} + \hat{b}(\vec{x}), \qquad \hat{u}(0) = \hat{u}_0 \tag{6.5}$$

Where $\hat{A} \in \mathbb{C}^{n \times n}$, $\hat{b} \in \mathbb{C}^{n}$, and $u \in \mathbb{C}^{n}$. Importantly, while solving the full-order model typically incurs a computational complexity of $O(N^3)$, the ROM significantly reduces this burden to is $O(n^3)$ for $n << N$. since the ROM is solved, one can recover the full-space solution by performing the inverse projection.

The key distinction with the PMM approach is that, unlike POD, which requires full knowledge of the high-fidelity operators. The PMM, an implicit-RBM method, only necessitates an understanding of the form of the governing equations, namely how each operator depends on the model parameters and how these operators interact. In the PMM framework, we still follow the same reduction strategy as in POD; however, the elements of the reduced operators $\hat{A}$,$\hat{b}$, and initial condition $\hat{u}_0$ are not predetermined. Instead, they are treated as trainable parameters (denoted by

$\underline{A}$,$\underline{b}$, and $\underline{u_0}$) that are optimized via gradient descent. The PMM for the above example is then written as

$$\frac{d\hat{u}}{dt} = \underline{A}(\vec{x})\hat{u} + \underline{b}(\vec{x}), \qquad \underline{u}(0) = \underline{u}_0 \tag{6.6}$$

After introducing the simple PMM example for the linear differential equation, we now consider how to handle nonlinear terms that arise in the full-order model. Suppose the governing equation includes a nonlinear term u^2, this operation corresponds to the Hadamard product $u \odot u$ i.e. elementwise square. In the standard POD approach, when a nonlinear operation is present, the reduced order model (ROM) must be temporarily lifted back to the full space to perform the nonlinearity before being re-projected onto the reduced subspace. Specifically, this process is represented as

$$u \odot u \overset{POD}{\longrightarrow} = P^{\dagger}[(P\hat{u}) \odot (P\hat{u})] \tag{6.7}$$

To avoid repeated lifting and projection during nonlinear evaluations, the projectors can instead be grouped and precomputed into a tensor representation. In index notation, this can be written as

$$\begin{aligned}(\hat{u} \odot \hat{u})_i &= P^{\dagger}_{iM} P_{Nj} \hat{u}_j P_{Nk} \hat{u}_k \\ &= [P^{\dagger}_{iN} P_{Nj} P_{Nk}] \hat{u_j} \hat{u_k} \\ &= T_{ijk} \hat{u}_j \hat{u}_k\end{aligned} \tag{6.8}$$

where $T_{ijk} \in \mathbb{C}^{n \times n \times n}$ is a precomputed nonlinearity tensor that captures the action of the full-space nonlinear operator in the reduced basis. Here, indices i, j, k correspond to the reduced space, while N runs over the full space. This tensor can be precomputed at a cost of $O(nN^3)$ remains efficient when $n << N$. This approach readily extends to polynomial nonlinearities of degree p resulting in a rank-$p+1$ tensor T. However, for $p > 2$ the precomputation cost becomes prohibitive, potentially dominating the overall computational expense of solving the ROM. In contrast, the PMM framework offers a significant advantage by learning the nonlinearity tensor $\underline{T}$ directly from data using gradient [22]. This bypasses the need for expensive precomputation, regardless of the degree p of the nonlinearity, and allows for efficient handling of complex nonlinear operations within the reduced order model framework.

Moreover, insights from tensor network methods (see Chapter 5) can be leveraged to design structured or low-rank ansätze for $\underline{T}$ Such tensor network inspired parameterizations e.g., matrix product state (MPS), Tensor Tree, Hyper-networks, or MERA forms, offer a principled way to initialize and constrain the nonlinearity tensor, enabling more efficient learning, improved interpretability, and reduced memory cost.

By leveraging the structure provided by the underlying physics, PMMs combine the rigor of classical model reduction techniques with the flexibility of machine learning, offering a promising pathway for efficiently and interpretably approximating high-dimensional, parameter-dependent systems.

This approach assumes that the parameter dependence of A, b, and T is either known exactly or approximated sufficiently well. In Section 6.7, we will extend this framework to a broader class of machine learning problems where the governing equations are not fully known, employing an ansätze inspired by the quantum mechanics observable formalism to tackle regression and classification tasks.

6.3 Hamiltonian Simulation

In chiral effective field theory (χEFT), the nuclear Hamiltonian takes the form

$$H = \sum_i T_i + \sum_{i<j} V_{ij} + \sum_{i<j<k} V_{ijk}, \tag{6.9}$$

where T_i denotes the kinetic energy of the i^{th} nucleon, V_{ij} is the two-body interaction, and V_{ijk} is the three-body force. The Hamiltonian is parameterized by low-energy constants (LECs), and can be expressed as a non-affine function $H(\text{LEC})$.

In realistic ab initio calculations, H is typically of such high dimension that direct diagonalization is computationally intractable, the Hamiltonian matrix, or even its eigenvectors, may not fit into memory. Sophisticated methods such as the In-Medium Similarity Renormalization Group (IMSRG), Coupled Cluster (CC), or Lattice EFT can be employed to approximate the ground-state energy of H, but these methods themselves are computationally expensive (see Chapter 2). As a result, only a limited number of ground-state energy evaluations $E(\text{LEC})$ may be available, often

with noise and possibly only accurate to first-order perturbation theory. This scarcity of accurate, full-scale simulations poses a significant challenge, especially in uncertainty quantification or parameter inference tasks, where billions of Hamiltonian evaluations might be required.

To address this, surrogate modeling techniques, such as eigenvector continuation (EC) [30] and related subspace projection methods [66], have been developed. These approaches project the Hamiltonian onto a low-dimensional subspace then solving a generalized eigenvalue problem. However, these methods generally require access to the eigenvectors of H to construct a reduced-order model (ROM). Experimental observations and frameworks like IMSRG, the eigenvectors are not available, and in others, they may be too large to store in memory.

This is where the Parametric Matrix Model (PMM) offers a new avenue. Instead of requiring access to the full Hamiltonian or its eigenvectors, PMMs only require scalar outputs—e.g., the ground-state energy $E(\vec{c})$ or expectation values of observables—as functions of the model parameters $\vec{c}$. These scalar quantities are readily produced by virtually all many-body methods.

6.3.1 Abstract Example: Affine Hamiltonian

To illustrate this idea, we simplify the setup and consider an abstract affine Hamiltonian of the form

$$H(\vec{c}) = H_0 + \sum_{i=1}^{k} c_i H_i, \tag{6.10}$$

where $H_i \in \mathbb{C}^{N\times N}$ are Hermitian operators and $\vec{c} \in \mathbb{R}^k$ are tunable parameters (e.g., LECs).

Following the POD approach described in Section 6.2, suppose we had access to a set of ground-state wavefunctions $|\psi(\vec{c}_1)\rangle, \ldots, |\psi(\vec{c}_M)\rangle$. We could then construct a data matrix

$$|\Psi(\vec{c})\rangle = [|\psi(\vec{c}_1)\rangle \cdots |\psi(\vec{c}_M)\rangle \tag{6.11}$$

and perform singular value decomposition (SVD) to extract a low-dimensional subspace spanned by the top n singular vectors. Projecting $H(\vec{c})$ into this subspace yields a reduced Hamiltonian

$$M(\vec{c}) = M_0 + \sum_{i=1}^{k} c_i M_i, \quad M_i = P^\dagger H_i P, \tag{6.12}$$

where $P \in \mathbb{C}^{N\times n}$ is the projection matrix, and $M_i \in \mathbb{C}^{n\times n}$ are the reduced operators. This procedure is equivalent to what is done in eigenvector continuation.

However, unlike EC, the PMM does not require wavefunctions. Instead, the above construction serves only to motivate the structure of the PMM ansätze:

$$M(\vec{c}) = \underline{M}_0 + \sum_{i=1}^{k} c_i \underline{M}_i, \tag{6.13}$$

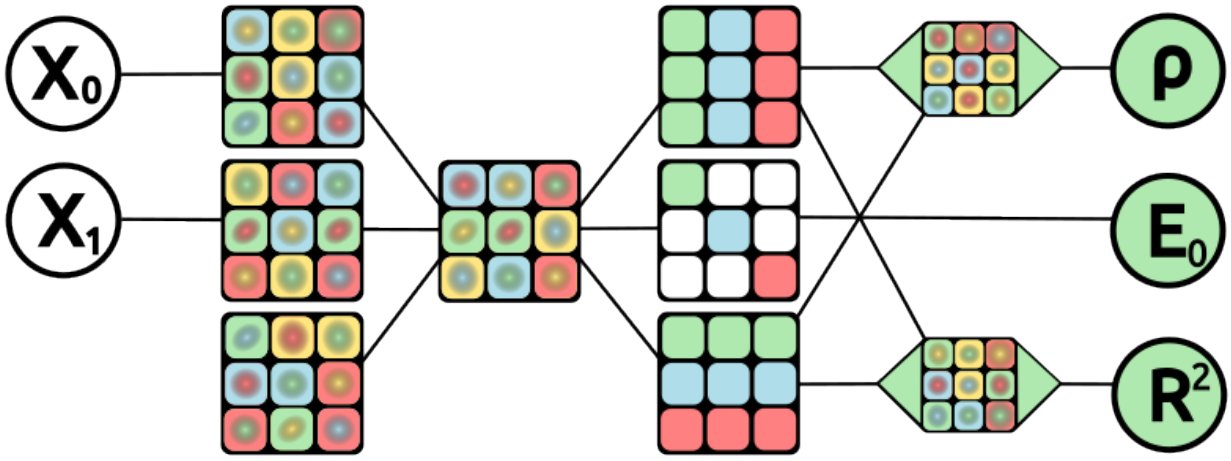

Figure 6.1 Diagrammatic illustration of the Hamiltonian simulation PMM architecture.

where $\underline{M}_i \in \mathbb{C}^{n \times n}$ are now trainable matrices, learned directly from scalar data (e.g., ground-state energies).

Given a dataset of ground-state energies $E(\vec{c}i)_{i=1}^{M}$ corresponding to various parameter configurations, we define the loss function:

$$\mathcal{L} = \frac{1}{M} \sum_{i=1}^{M} \left(\epsilon(\vec{c}_i) - E(\vec{c}_i)\right)^2, \tag{6.14}$$

where $\epsilon(\vec{c}_i)$ is the smallest eigenvalue of the PMM matrix $M(\vec{c}_i)$ in (6.13), computed via exact diagonalization (since $n << N$). The PMM matrices $\underline{M}_i$ are optimized using gradient descent to minimize this loss.

The PMM framework can also accommodate the learning of observables. Suppose we have access to scalar measurements of an observable O in the ground state of H, i.e., $\langle \psi(\vec{c}) | O | \psi(\vec{c}) \rangle$.

We may learn a reduced observable $\underline{\Delta}$ such that:

$$\langle\phi(\vec{c})|\underline{\Delta}|\phi(\vec{c})\rangle \approx \langle\psi(\vec{c})|O|\psi(\vec{c})\rangle, \tag{6.15}$$

where $|\phi(\vec{c})\rangle$ is the ground state of the PMM matrix $M(\vec{c})$. Figure 6.1, showcases a diagrammatic representation of PMM architecture employed in this example.

By learning reduced operators directly from observable data, PMMs circumvent the need for storing or computing wavefunctions. This capability opens a promising pathway toward efficient, interpretable, and data-efficient surrogate models for nuclear physics, particularly in regimes where only scalar observables from either experiments or large scale *ab initio* many-body solvers are available.

6.3.2 Results

In this section, we present several examples that demonstrate the performance and versatility of the PMM framework when applied to various Hamiltonian problems. For clarity, we focus on two illustrative cases: one comparing PMM with eigenvector continuation (EC) for a simple Hamiltonian, and another vs other ML methods using the Anharmonic Lipkin-Meshkov-Glick (ALMG) model, a more challenging system. Additional examples are briefly highlighted later at the end of the section.

6.3.2.1 Comparison with EC

We first consider the eigenvalue problem for a family of Hermitian matrices, $H(\vec{c})$, whose elements are analytic functions of real parameters $\vec{c}$. This is a natural setting for reduced basis methods like eigenvector continuation (EC). In EC, one selects a set of eigenvector snapshots at training points,

$$[\vec{c}_1, \vec{c}_2, \cdots, \vec{c}_M] \tag{6.16}$$

and projects the full Hamiltonian onto the subspace spanned by these snapshots, is then diagonalized to obtain the eigenvalues and eigenvectors. While effective for many problems, EC can suffer when the number of degrees of freedom is large, the number of snapshots required to maintain accuracy may scale unfavorably. The PMM approach reproduces the structure of EC by learning the

elements of the reduced matrices $M(\vec{c})$ directly. In the following, we present an example where PMMs perform better than EC for a problem with many degrees of freedom. We consider a simple system composed of N non-interacting spin-$\frac{1}{2}$ particles with the one-parameter Hamiltonian

$$H(c) = \frac{1}{2N} \sum_{i}^{N} (\sigma_i^z + c\sigma_i^x) \tag{6.17}$$

Here, σ_i^z and σ_i^x are the z and x Pauli matrices for spin i, respectively. In Fig. 6.2, we see that the EC method, using only five training snapshots obtained from the full N-spin problem, struggles to accurately reproduce the ground-state energy E_0 for large N. This degradation in performance arises because, as the dimensionality of the Hamiltonian increases, the subspace spanned by a limited number of eigenvector snapshots may fail to capture the relevant variation in the system's state. In such cases, the eigenvectors can rotate significantly as a function of the control parameter c, even when the corresponding eigenvalues remain nearly invariant. This rotation leads to a poorly conditioned snapshot subspace and limits the predictive accuracy of EC. In contrast, the PMM formulation successfully recovers E_0 exactly, using a learned 2×2 model of the form

$$M(c) = \frac{1}{2}(\sigma^z + c\sigma^x) \tag{6.18}$$

This example, though simple, highlights the increased flexibility of PMMs. Unlike EC, which is restricted to a fixed projected subspace, the PMM can adaptively learn an optimal latent representation (basis) directly from scalar observables, thereby circumventing the geometric limitations that hinder EC and overcoming the associated extrapolation challenges.

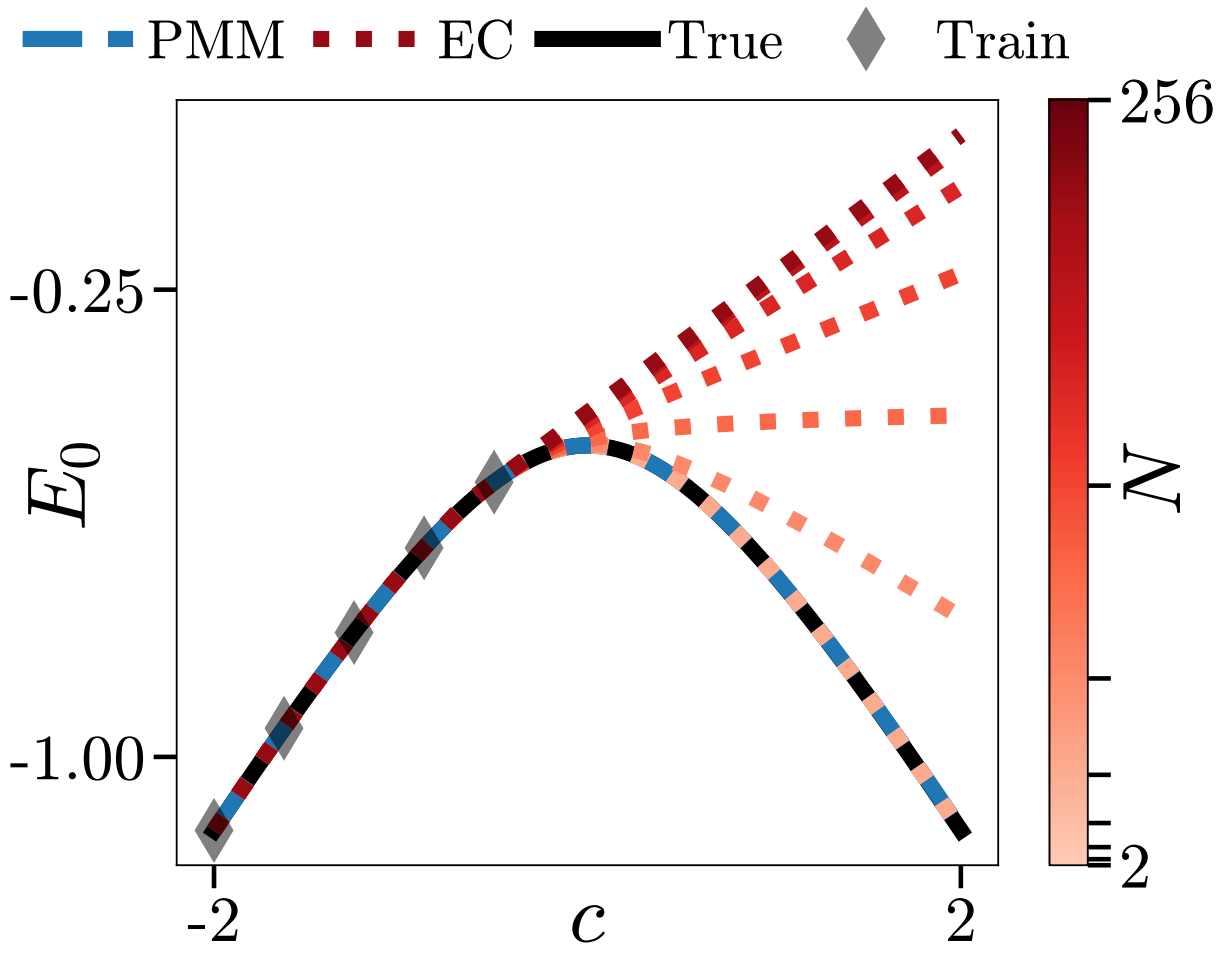

Figure 6.2 We show results for a 2×2 PMM (dashed blue) and EC (dotted red) with 5 training samples on the task of extrapolating the ground state energy of a system of N non-interacting spins. The exact ground state energy is shown in solid black. Figure taken from [23].

6.3.2.2 Anharmonic Lipkin Model

Next, we examine the more complex ALMG model, a quantum Hamiltonian describing-$\frac{1}{2}$ particles on N lattice sites with tunable anharmonic long-range two-body interactions. The model is defined as

$$H(\xi;\alpha) = (1-\xi)\hat{n} + \frac{2\xi}{S}(S^2 - \hat{S}_x^2) + \frac{\alpha}{2S}\hat{n}(\hat{n}+1)$$
$$\hat{n} = (S + \hat{S}_z) \tag{6.19}$$

with the anharmonicity parameter set to $\alpha = -0.6$. The ALMG model exhibits a second-order ground-state quantum phase transition as a function of the parameter ξ (specifically, near $\xi = 0.2$),

where in the large N limit, the ground state becomes degenerate and the average particle density $\langle \hat{n} \rangle / N$, becomes nonzero, further details in Ref [38, 31]. Here, we consider the case $N = 1000$. By reorganizing the Hamiltonian, one can cast it in an affine form

$$H_\alpha(\xi) = H_0 + \xi H_I \tag{6.20}$$

From data consisting of ground-state energies $E(\xi)$ and expectation values $\langle \psi_\alpha(\xi) | \hat{n} | \psi_\alpha(\xi) \rangle$ in the ground state of $H_\alpha(\xi)$, the PMM constructs a corresponding reduced mode

$$\begin{aligned} M_\alpha(\xi) =& \underline{M}_0 + \xi \underline{M}_I \\ & \langle \phi_\alpha(\xi) | \underline{\Delta} | \phi_\alpha(\xi) \rangle \end{aligned} \tag{6.21}$$

where $\underline{M_i}$ and $\underline{\Delta}$ are the trainable matrices corresponding to the effective Hamiltonian and observable, respectively. In our implementation, we use a 9×9 PMM to represent these effective operators, and we optimize the model by minimizing the mean squared error between the eigenvalues of $M_\alpha(\xi)$ and the provided energy data, as well as between the effective observable and the measured particle density, further detail can be fond in Ref [23].

Figure 6.3 compares the PMM predictions with those from a multilayer perceptron (MLP) for both the lowest two energy levels and the average particle density, $\langle \hat{n} \rangle / N$, as a function of ξ. In both cases, the PMM significantly outperforms the MLP, accurately capturing the phase transition behavior. This is particularly evident in the derivative of the ground-state energy, $dE_0(\xi)/d\xi$, where the MLP exhibits spurious oscillations, whereas the PMM provides a smooth and physically consistent representation.

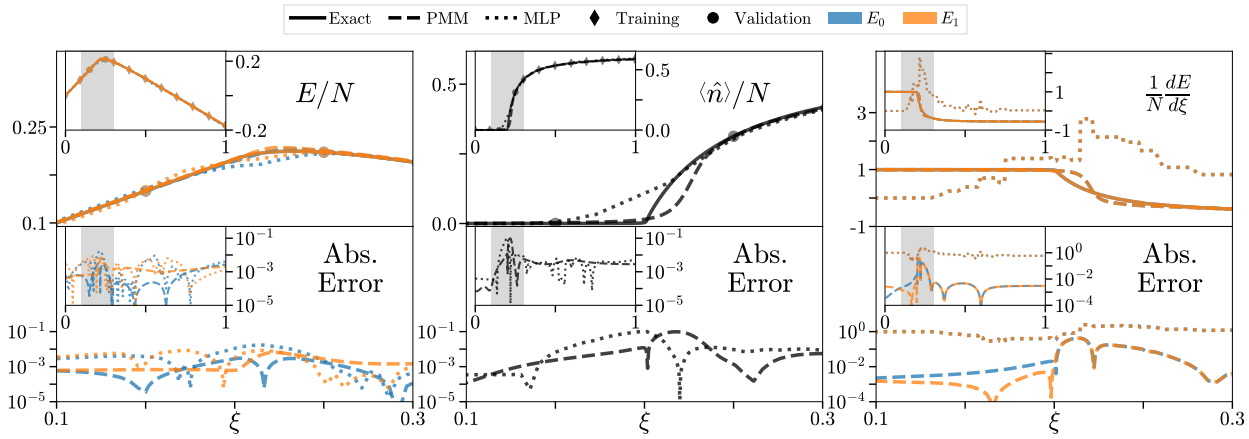

Figure 6.3 Left Panel: Lowest two energy levels of the ALMG model versus ξ. We show PMM results compared with Multilayer Perceptron (MLP) results. The upper plots show the energies, and the lower plots show absolute error. The main plots show the region around the phase transition; the insets show the full domain where data was provided. Center Panel: Average particle density for the ground state of the ALMG model versus ξ. We show PMM results compared with Multilayer Perceptron (MLP) results. The upper plots show the average particle density, and the lower plots show absolute error. The main plots show the region around the phase transition; the insets show the full domain where data was provided. Right Panel: Derivative of the lowest two energy levels with respect to the control parameter ξ as a function of ξ. We show PMM results compared with Multilayer Perceptron (MLP) results. The upper plots show the derivatives of the energies with respect to the control parameter, and the lower plots show absolute error. The main plots show the region around the phase transition; the insets show the full domain where data was provided. No data on the derivatives was provided to either model. Figure taken from [23]

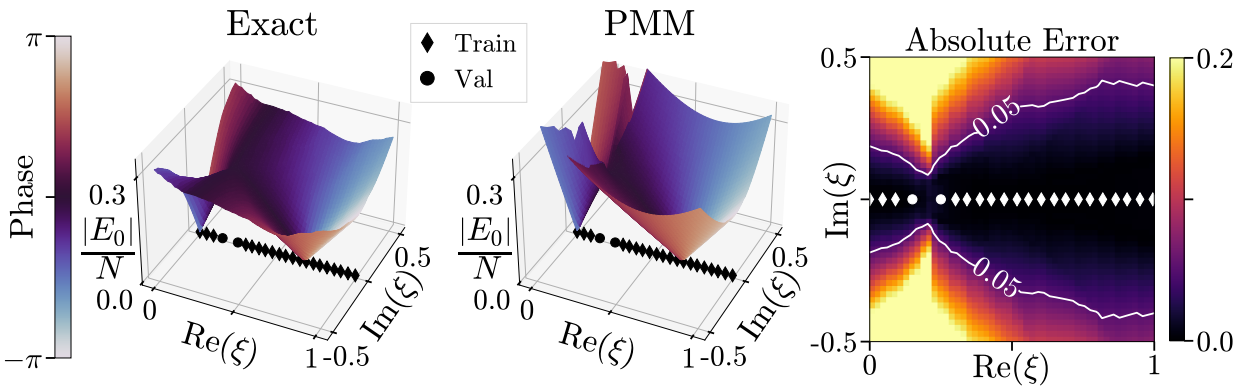

Figure 6.4 We show PMM predictions for the complex-valued ground state energy for complex values of ξ, using training data at only real values of ξ. The left plot shows the exact results, the middle plot shows the PMM predictions, and the right plot shows the absolute error. Figure taken from [23]

Moreover, the PMM framework offers a novel advantage: its analytically grounded design allows for the extrapolation of the system's behavior into the complex plane without extra training. Figure 6.4 demonstrates this capability by comparing the complex-valued ground-state energy predictions from the trained PMM with the exact results. Despite having been trained solely on real ξ data, the PMM accurately reproduces the expected behavior, including the locations of exceptional points and avoided level crossings, with errors below 0.05 in key regions. This capability not only demonstrates the robustness of the PMM approach but also unlocks new opportunities for model discovery by enabling the identification of complex branching structures and phase transitions.

Further examples in our work include the Anharmonic Oscillator and extended analyses of the Lipkin model for multiple states and observables (see Figs. 6.5 and **??**). These cases underscore the PMM's robustness in modeling Hamiltonian energies and capturing sharp discontinuities, an aspect where traditional machine learning methods or standard interpolation techniques (like cubic splines) typically fail.

6.3.3 Summary

Since its introduction [23], the Parametric Matrix Model (PMM) framework has demonstrated broad applicability and has been rapidly adopted by multiple research groups. PMMs provide a

data-efficient and interpretable approach for constructing reduced-order representations of high-dimensional Hamiltonians, relying solely on scalar observables rather than full wavefunctions. Applications include building fast emulators for Bayesian inference [82, 2, 25], modeling the equation of state for infinite neutron matter [83] using observational data from LIGO and NASA NICER, exploring ground-state energies in recently discovered nuclei, such as ^{28}O [98], using IMSRG-generated energy data, studying neutron and Fermi polarons using auxiliary-field Quantum Monte Carlo methods on the lattice [26], and modeling Linear response for quasiparticle random-phase approximation (QRPA) [41] .

These studies highlight several key advantages of PMMs: they circumvent the storage and computational challenges associated with traditional many-body methods, can accurately capture phase transitions and sharp energy discontinuities, and provide robust extrapolation capabilities, including into the complex parameter plane. Furthermore, the flexibility to learn both effective Hamiltonians and observables enables PMMs to outperform standard machine learning models and interpolation techniques. Collectively, these results underscore the PMM framework as a powerful, generalizable tool for efficient simulation, analysis, and discovery in nuclear many-body physics.

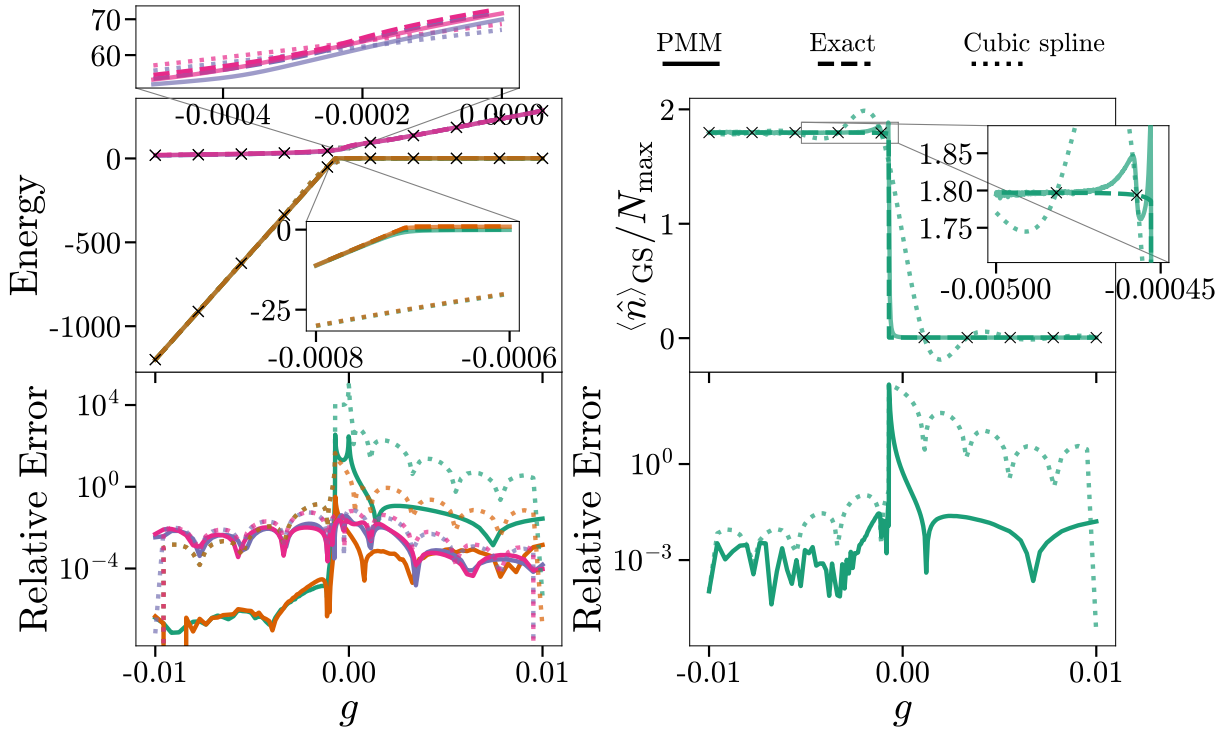

Figure 6.5 Comparison between the PMM (solid line) and cubic spline (dotted line) models against the true observables of the anharmonic oscillator, where "×" marks the training data. Left: The four lowest energy levels, each shown in a different color. Right: The order parameter. The PMM successfully captures the first-order phase transition, whereas the cubic spline fails to reproduce this behavior.

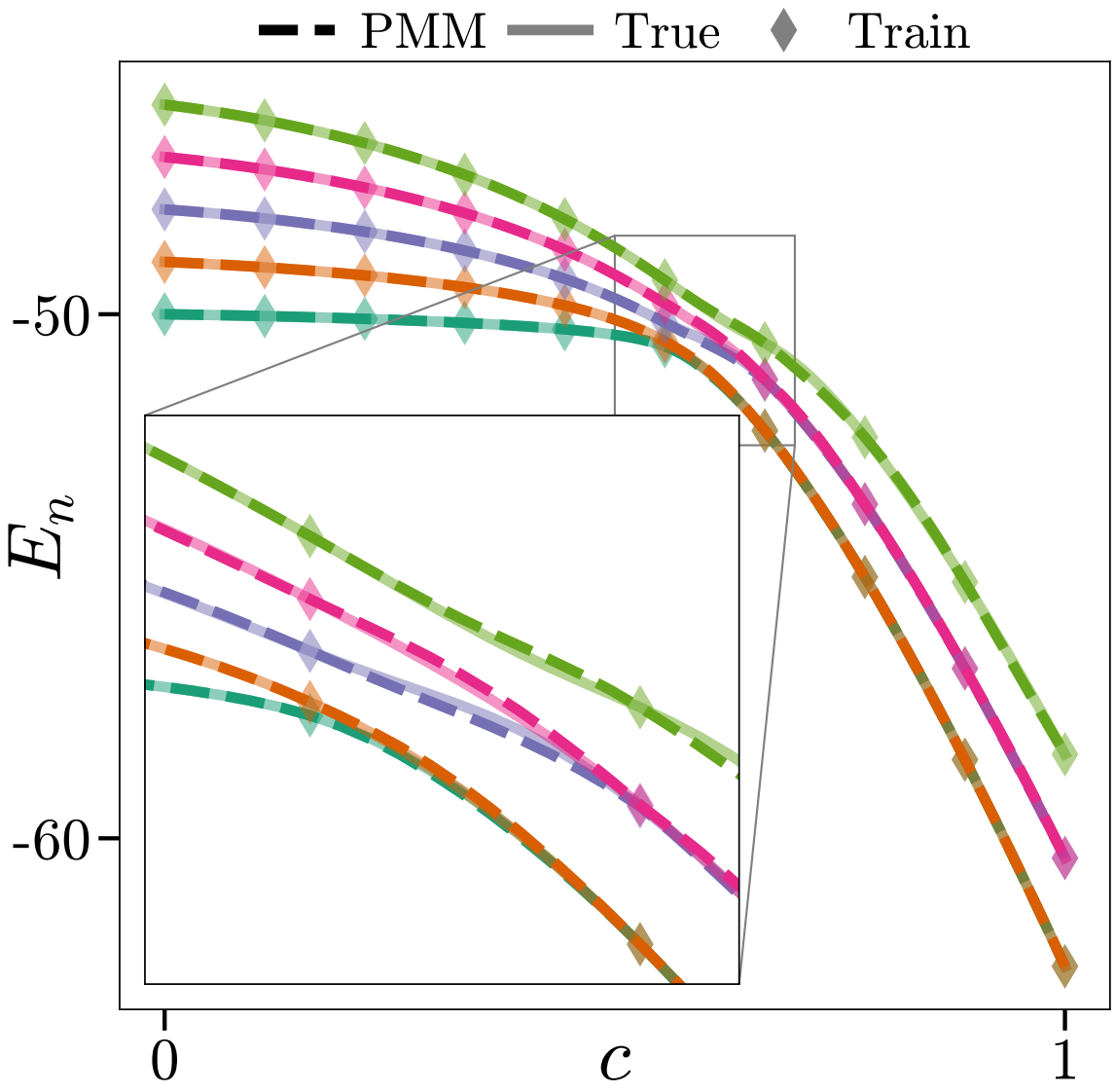

Figure 6.6 Results of training a 15×15 PMM on the five lowest eneriges of a 100 site LMG model in the region of the phase transition.

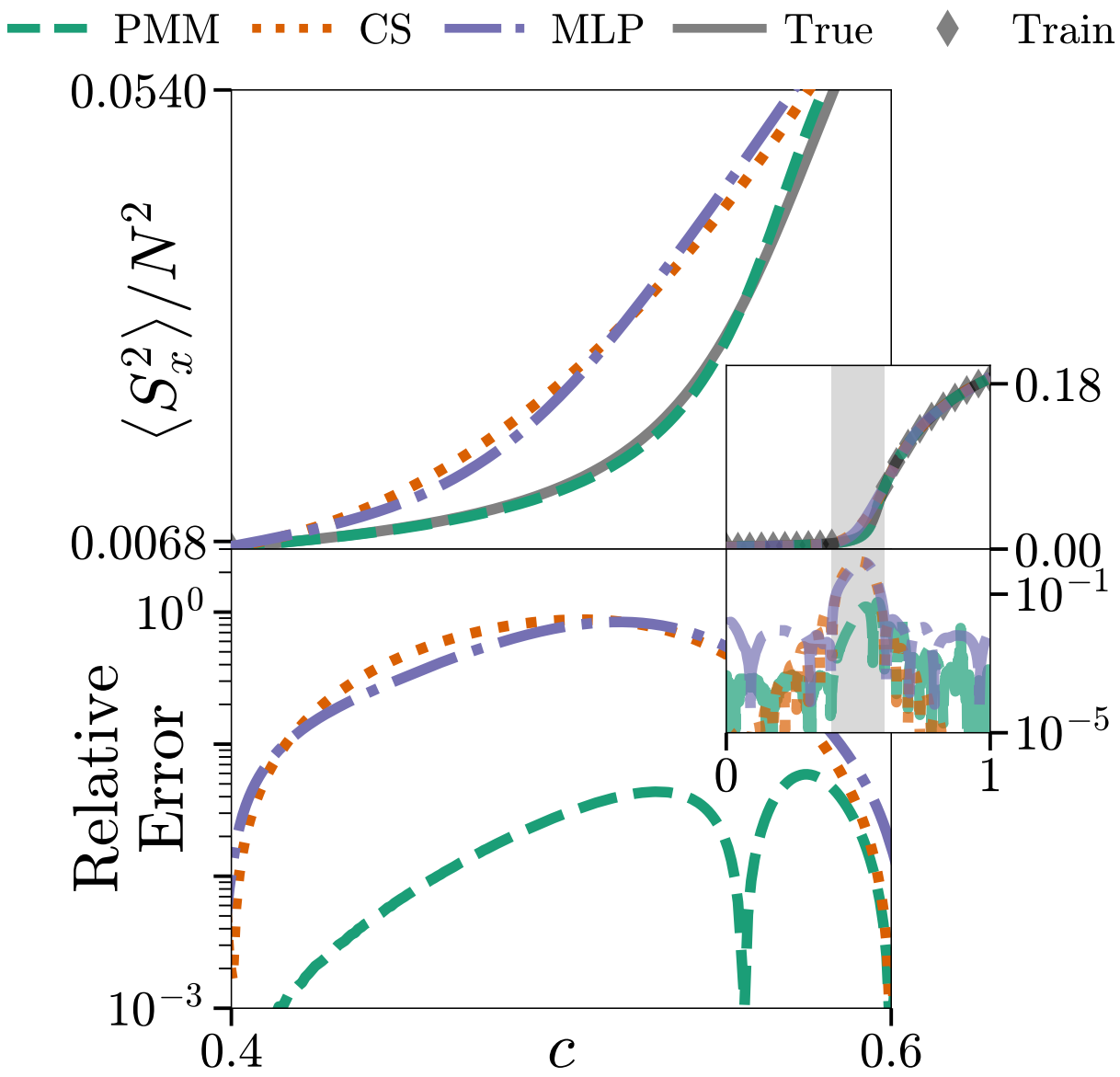

Figure 6.7 Results of using the ground state eigenvector of the PMM used to learn the energies of the LMG model to learn $\langle S_x^2 \rangle / N^2$ in the ground state. Comparisons with the average of 100 MLPs (dash-dotted) and cubic spline interpolation (dotted) show the superior performance of the PMM. The top plots show the learned (dashed) and exact (solid) values; the bottom plots show the relative error. The main figure focuses in on the region of the phase transition, where there was no training data. The inset shows the full domain $0 \leq c \leq 1$ with the focused region highlighted.

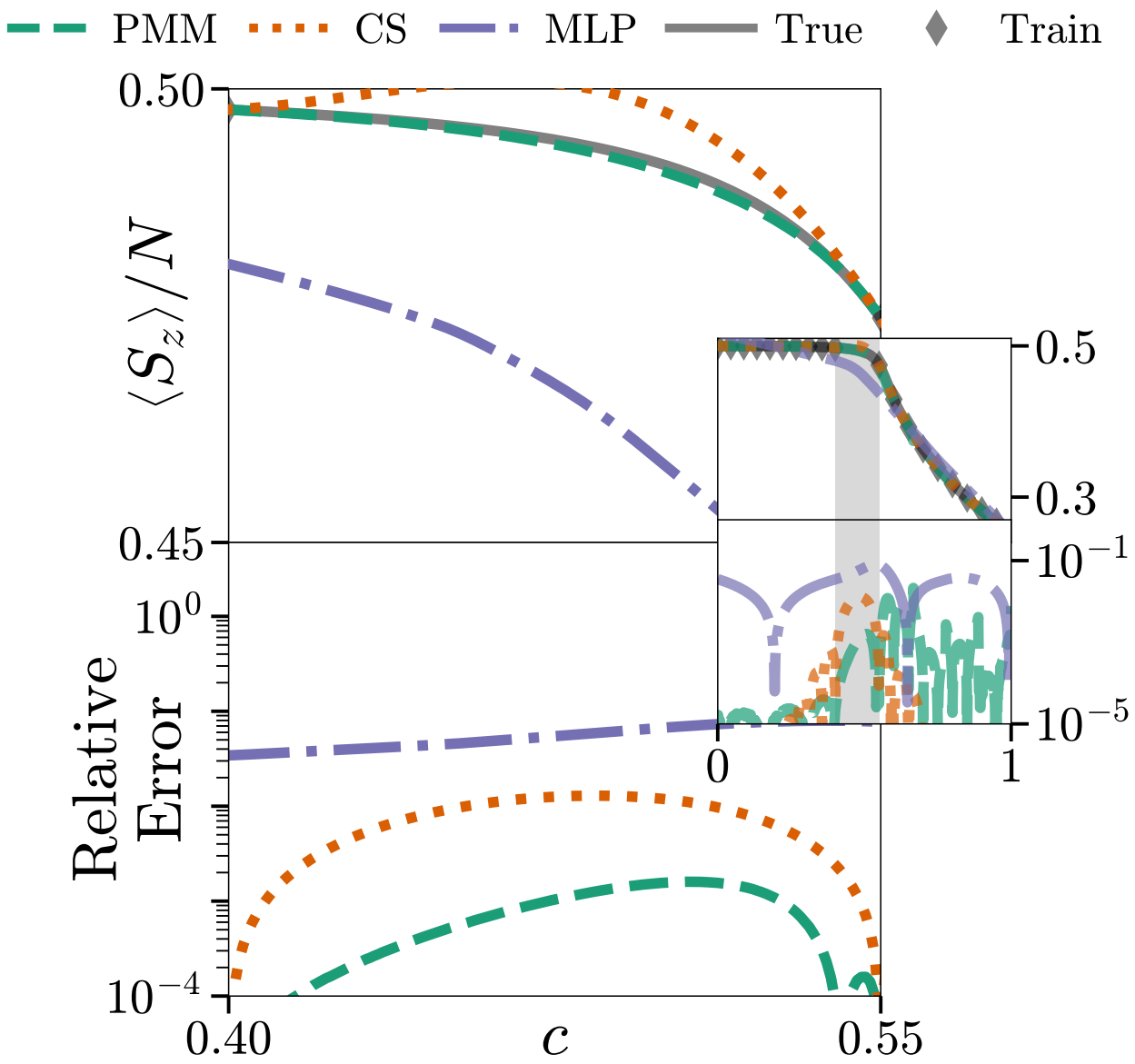

Figure 6.8 Results of using the ground state eigenvector of the PMM used to learn the energies of the LMG model to learn $\langle S_z^2 \rangle / N^2$ in the ground state. Comparisons with the average of 100 MLPs (dash-dotted) and cubic spline interpolation (dotted) show the superior performance of the PMM. The top plots show the learned (dashed) and exact (solid) values; the bottom plots show the relative error. The main figure focuses in on the region of the phase transition, where there was no training data. The inset shows the full domain $0 \leq c \leq 1$ with the focused region highlighted.

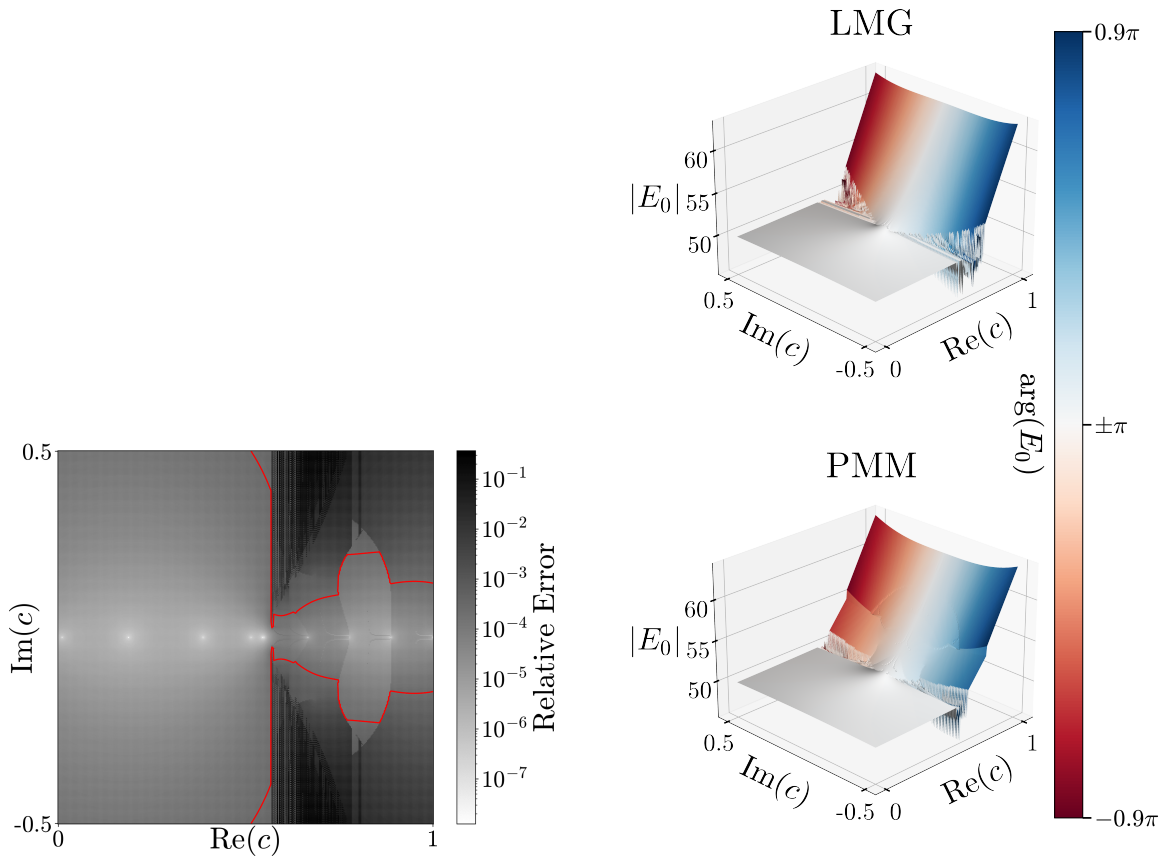

Figure 6.9 Results of using the PMM trained on energies of the LMG model with real-valued c to extrapolate the ground state energy for complex valued c. The left plots show the learned (bottom) and exact (top) values. The height of the surface corresponds to the magnitude of E0 while the color corresponds to the argument. The right plot shows the relative error between the exact values and the learned extrapolated values. The red contour shows where the error is 10^3.

6.4 Euclidean Time Evolution

As discussed in Section 6.3, one of the ab initio approaches for solving the many-body χEFT Hamiltonian is lattice effective field theory (lattice EFT). In lattice EFT, a common strategy for obtaining the ground state of the Hamiltonian is to perform Monte Carlo simulations of the Euclidean-time evolution operator. A major computational challenge in this framework is the sign problem, which arises because the many-body wavefunction, or equivalently the path-integral weight, can take on negative or complex values. Since Monte Carlo methods rely on sampling from positive-definite probability distributions, negative amplitudes cannot be directly interpreted as probabilities. In practice, one samples using the absolute value of the weight and reweights by its sign or phase, but this reweighting leads to severe cancellations and exponentially growing statistical noise with system size or Euclidean time. These cancellations produce exponentially large statistical uncertainties as the system size or evolution time increases, rendering straightforward Monte Carlo sampling ineffective.

Typically, the Hamiltonian is partitioned as

$$H = H_0 + cH_I, \tag{6.22}$$

where H_0 is the "easy" unperturbed Hamiltonian that does not suffer from the sign problem, and H_I is the perturbative interaction whose contributions can induce sign oscillations, with c controlling the interaction strength. As c or the system size increases, the severity of the sign problem grows, dramatically increasing the number of Monte Carlo samples required for reliable convergence.

Due to this difficulty, practical lattice EFT calculations often resort to truncating the computation at first-order perturbation theory in H_I which limits the accuracy of the resulting ground-state energies and observable expectation values. Convergence beyond first order is generally inaccessible in large systems, leaving a sparse and approximate dataset for downstream analyses.

This is precisely where the Parametric Matrix Model (PMM) framework provides a useful advantage. Even with access only to first-order perturbative energies or observables, PMMs can learn a latent space representation of the Hamiltonian and relevant operators. By optimizing the reduced matrices directly against these limited scalar outputs, PMMs effectively resum the

perturbation series, constructing a surrogate model that captures nonperturbative behavior and enables accurate predictions across the full parameter range, including regimes that are otherwise inaccessible due to the sign problem.

6.4.1 Theory

We begin by defining imaginary-time evolution, which forms the basis of many ab initio lattice simulations in nuclear and condensed matter physics. Starting from an initial state $|\psi(0)\rangle$, the Euclidean (imaginary-time) evolution is given by

$$|\psi(\tau)\rangle = e^{-H\,\tau/\hbar}\,|\psi(0)\rangle, \quad \tau \in \mathbb{R}. \tag{6.23}$$

Expanding the initial state in the energy eigenbasis of H, we obtain

$$|\psi(\tau)\rangle = \sum_n c_n\, e^{-E_n\,\tau/\hbar}\,|E_n\rangle. \tag{6.24}$$

As τ increases, higher-energy components are exponentially suppressed relative to the ground state. In the asymptotic limit, the system thus projects onto $|E_0\rangle$, providing a direct path to extract the ground-state energy.

In lattice effective field theory (EFT), the Hamiltonian is typically decomposed into an exactly solvable part and an interaction term that exhibit a MC sign problem,

$$H = H_0 + \lambda H_I, \tag{6.25}$$

where H_0 captures the unperturbed dynamics and H_I introduces corrections arising from interactions [56]. The ground-state energy and observables are estimated from imaginary-time correlation functions. For example, the unperturbed ground-state energy is

$$E^{(0)} = \frac{\langle\psi|e^{-H_0\tau}H_0e^{-H_0\tau}|\psi\rangle}{\langle\psi|e^{-H_0\tau}|\psi\rangle}, \tag{6.26}$$

where $Z(\tau) = \langle\psi|e^{-H_0\tau}|\psi\rangle$ is the partition function. The first-order correction due to H_I can similarly be written as

$$E^{(1)} = \frac{\langle\psi|e^{-H_0\tau}H_Ie^{-H_0\tau}|\psi\rangle}{\langle\psi|e^{-H_0\tau}|\psi\rangle}. \tag{6.27}$$

In practice, however, lattice Monte Carlo (MC) calculations suffer from the sign problem: when amplitudes in the path integral acquire alternating signs or complex phases, large cancellations occur. This leads to exponentially growing statistical noise with increasing system size or τ, severely restricting direct MC access to large imaginary times. Consequently, calculations are often limited to short-time data or low-order perturbation theory.

Consider MC data for the Euclidean-time evolution, as schematically shown in Fig. **??**. To extract the energy as a function of imaginary time, we can use the logarithmic derivative of the partition function,

$$E(\tau) = -\frac{\hbar}{2}\frac{1}{Z(\tau)}\frac{dZ}{d\tau}. \tag{6.28}$$

Substituting the spectral expansion of Z, we obtain

$$E(\tau) = \frac{\sum_n |c_n|^2 E_n e^{-2E_n\tau/\hbar}}{\sum_n |c_n|^2 e^{-2E_n\tau/\hbar}}. \tag{6.29}$$

At large τ, only the lowest few states survive, and to leading order we can approximate

$$E(\tau) \approx E_0 + (E_1 - E_0)\frac{|c_1|^2}{|c_0|^2}e^{-2(E_1-E_0)\tau/\hbar}. \tag{6.30}$$

This motivates an exponential fit of the form

$$f(\tau) = A + Be^{-\tau/\tau_0}, \tag{6.31}$$

as illustrated by the orange curve in Fig. **??**. Extrapolating $f(\tau)$ to large τ yields an estimate of the ground-state energy, which can then be used as a prediction for subsequent calculations or as high-quality training data for machine learning emulators.

Although early-time data are often discarded as "excited-state contamination" and used merely to stabilize exponential fits, they in fact encode valuable information about the higher-energy spectrum and matrix elements of the interaction. From many-body perturbation theory,

$$E_n(\lambda) = E_n^{(0)} + \lambda\langle\psi_n^{(0)}|H_I|\psi_n^{(0)}\rangle + \lambda^2\sum_{k\neq n}\frac{\left|\langle\psi_k^{(0)}|H_I|\psi_n^{(0)}\rangle\right|^2}{E_n^{(0)} - E_k^{(0)}} + O(\lambda^3), \tag{6.32}$$

$$|\psi_n(\lambda)\rangle = |\psi_n^{(0)}\rangle + \lambda\sum_{k\neq n}\frac{\langle\psi_k^{(0)}|H_I|\psi_n^{(0)}\rangle}{E_n^{(0)} - E_k^{(0)}}|\psi_k^{(0)}\rangle + O(\lambda^2), \tag{6.33}$$

showing that excited states contribute systematically to higher-order energy corrections. Hence, if a model can learn from short-time Euclidean evolution, it may implicitly reconstruct a *resummation* of the perturbative expansion by capturing correlations among early-time excitations.

In what follows, we seek to recover this spectral information through the Parametric Matrix Model (PMM) framework. Since the underlying equations are similar to the previous section, we assume that the PMM employs a similar ansätze. The PMM mirrors the structure of the underlying physical system by introducing learnable matrices that play the role of the Hamiltonian and its constituent components. Specifically, we define

$$|\underline{\phi}\rangle \in \mathbf{C}^n, \qquad \underline{M} = \underline{M_0} + c\,\underline{M_I}, \tag{6.34}$$

where $\underline{M_0}$ and $\underline{M_I}$ correspond respectively to the unperturbed and interaction components, while $|\underline{\phi}\rangle$ denotes the initial state. The model is initialized with random trainable parameters and optimized directly from Euclidean-time data. Analogously to the lattice EFT definitions, we define the effective PMM energies as

$$\epsilon^{(0)} = \frac{\langle\underline{\phi}|e^{-\underline{M_0}\tau}\,\underline{M_0}\,e^{-\underline{M_0}\tau}|\underline{\phi}\rangle}{\langle\underline{\phi}|e^{-\underline{M_0}\tau}|\underline{\phi}\rangle}, \tag{6.35}$$

$$\epsilon^{(1)} = \frac{\langle\underline{\phi}|e^{-\underline{M_0}\tau}\,\underline{M_I}\,e^{-\underline{M_0}\tau}|\underline{\phi}\rangle}{\langle\underline{\phi}|e^{-\underline{M_0}\tau}|\underline{\phi}\rangle}. \tag{6.36}$$

This formulation preserves the mathematical structure of the Euclidean-time evolution while replacing explicit Monte Carlo sampling with a differentiable parametric surrogate. By training on early-time Monte Carlo data, the PMM can, in principle, learn an effective latent representation that captures and resums higher-order perturbative effects, offering a data-driven route to surpass first-order perturbative limitations imposed by the sign problem.

6.4.2 Results

We demonstrate the performance of the PMM framework on Euclidean-time evolution data through two representative applications. The first is a proof-of-concept study based on synthetic lattice simulations, and the second involves real Monte Carlo lattice EFT (MCLEFT) data. Although

the latter is not derived from a fully *ab initio* Hamiltonian, it provides a meaningful benchmark for assessing the PMM's ability to handle the stochastic noise inherent in Monte Carlo calculations.

6.4.2.1 Two-Body Lattice Simulation

First, we consider a synthetic lattice simulation. We model a system of two charged particles on a three-dimensional lattice, governed by the Hamiltonian [54, 7]

$$H = KE + V_\delta + V_S(R_w) + cV_C(r) \tag{6.37}$$

Where KE is the kinetic energy operator, V_δ is a finite well of depth $-V0$ localized at the origin, $V_S(R_w)$ is a hard-sphere (infinite) wall of radius R_w, and $Vc(r)$ is an attractive coulomb potential acting between the two particles. The Hamiltonian is partitioned as

$$H = H_0 + cH_I, \tag{6.38}$$

where $H_0 = KE + V_\delta + V_S(R_w)$ defines the unperturbed component, and $H_I = V_C(r)$ introduces the perturbative interaction.

Our dataset consists of two-time, two-state Euclidean correlators $\mathcal{M}$ computed for multiple initial states of the unperturbed Hamiltonian H_0, together with their first-order corrections due to H_I, and measurements of relevant observables O (such as density or charge radius).

The correlator is defined as

$$\mathcal{M}_{ij}(t_1, t_2) = \langle\psi_i|e^{-H_0 t_1/2} O e^{-H_0 t_2/2}|\psi_j\rangle \tag{6.39}$$

for a family of initial states $|\psi_k\rangle$, and with the operator O taking each of the three roles

$$O \in \{H_0, H_I, \rho\} \tag{6.40}$$

.

The PMM is then trained to learn the effective operators $\underline{M}_0$, $\underline{M}_I$, $\underline{\Delta}_{\text{eff}}$, and the state $|\underline{\psi_i}\rangle$ by fitting the model-predicted correlators $\hat{\mathcal{M}}$ to the Euclidean correlators $\mathcal{M}$. Because the reference state is inaccessible, it is treated as a learnable quantity. Through this process, the PMM reconstructs an interpretable, reduced-order representation of the underlying dynamics.

Once trained, the reduced Hamiltonian

$$M(c) = \underline{M}_0 + c\underline{M_I} \tag{6.41}$$

is diagonalized to obtain its ground state $|\phi(c)\rangle$ and energy $\epsilon(c)$. The effective observable is then computed as

$$\langle\phi(c)|\underline{\Delta}_{eff}|\phi(c)\rangle \tag{6.42}$$

Figure 6.10a) compares the PMM's prediction of the ground-state energy of the full Hamiltonian H in both the bound- and unbound-H_0-cases. Although the PMM is trained only on first-order data from H_I, it resums the perturbation series and accurately extrapolates into the strong-coupling regime (c large), where the opposite spectral behavior emerges (an unbound state appears when H_0 is bound, and vice versa). These panels also showcase how the nonperturbative ground-state energy converges systematically as the PMM matrix size n is increased. In Figure 6.11 we train ten independent complex hermitian PMMs of size 9×9 (for the bound H_0 case) on the same first-order data. PMM component $\underline{M_I}$ is constrained to be negative definite, while $\langle\underline{\Delta}\rangle$ is constrained to have a spectral norm $||\underline{\Delta}|| \leq 1$, reflecting properties of the known operators in the original system. We plot statistics and error metrics for the ground-state energy, its derivative, and for other ground-state observables—density and charge radii— as functions of c, demonstrating that the PMM can also extrapolate these observables deep into the strong-coupling regime. To our knowledge, no existing method can perform such extrapolation using only Euclidean-time evolution data—data that contain no explicit information about how energies or observables vary with c. Even when the PMM predictions are not exact, they offer a valuable—and often the only—quantitative framework for extrapolation beyond the perturbative regime. Figure 6.12, further demonstrates the PMM's remarkable ability to continue this extrapolation into the complex-c plane, allowing for the identification of branch points and phase transitions in the system.

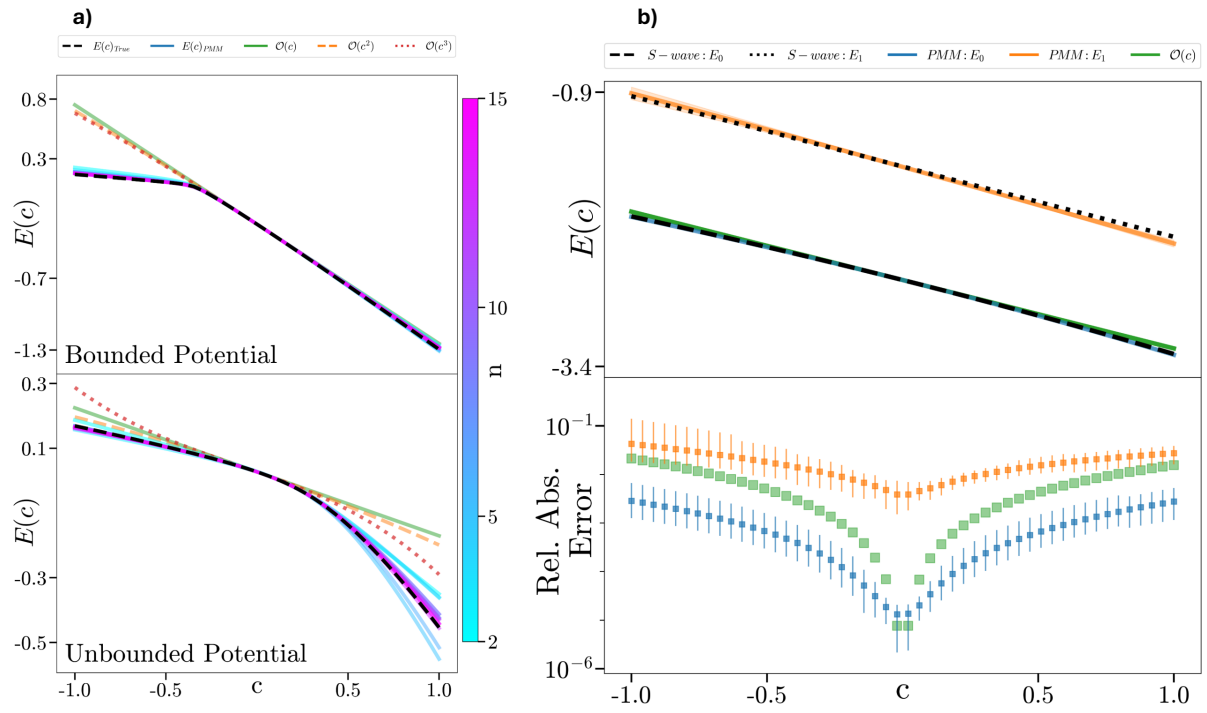

Figure 6.10 Two-body system on a three-dimensional lattice with periodic boundary conditions under an attractive Coulomb potential. **a)** The Hamiltonian includes a delta-function potential. We fit PMMs ranging from 2×2 to 15×15 to demonstrate convergence to the true (non-perturbative) ground state. *Top*: H_0 includes a bound state. *Bottom*: H_0 does not include a bound state. **b)** The Hamiltonian includes a spherical well. We fit a 9×9 PMM to predict both the ground and first excited state. $O(1)$, $O(2)$, and $O(3)$ indicates first-, second-, third-order corrections.

In the third simulation, we extend the system from a delta-function potential to a finite spherical well, which supports multiple bound states. We train a PMM to predict both the S-wave ground state and the first excited state sharing the same quantum number. Figure 6.10b) displays this comparison: the PMM accurately tracks both the ground and first excited energies as functions of c. Notably, the PMM error observed for the first excited state is on par with a the first-order correction for the ground state. Which creates a path for Lattice-EFT/QCD and Neural Quantum State (NQS) to predict excited-state behavior in regimes that were previously inaccessible.

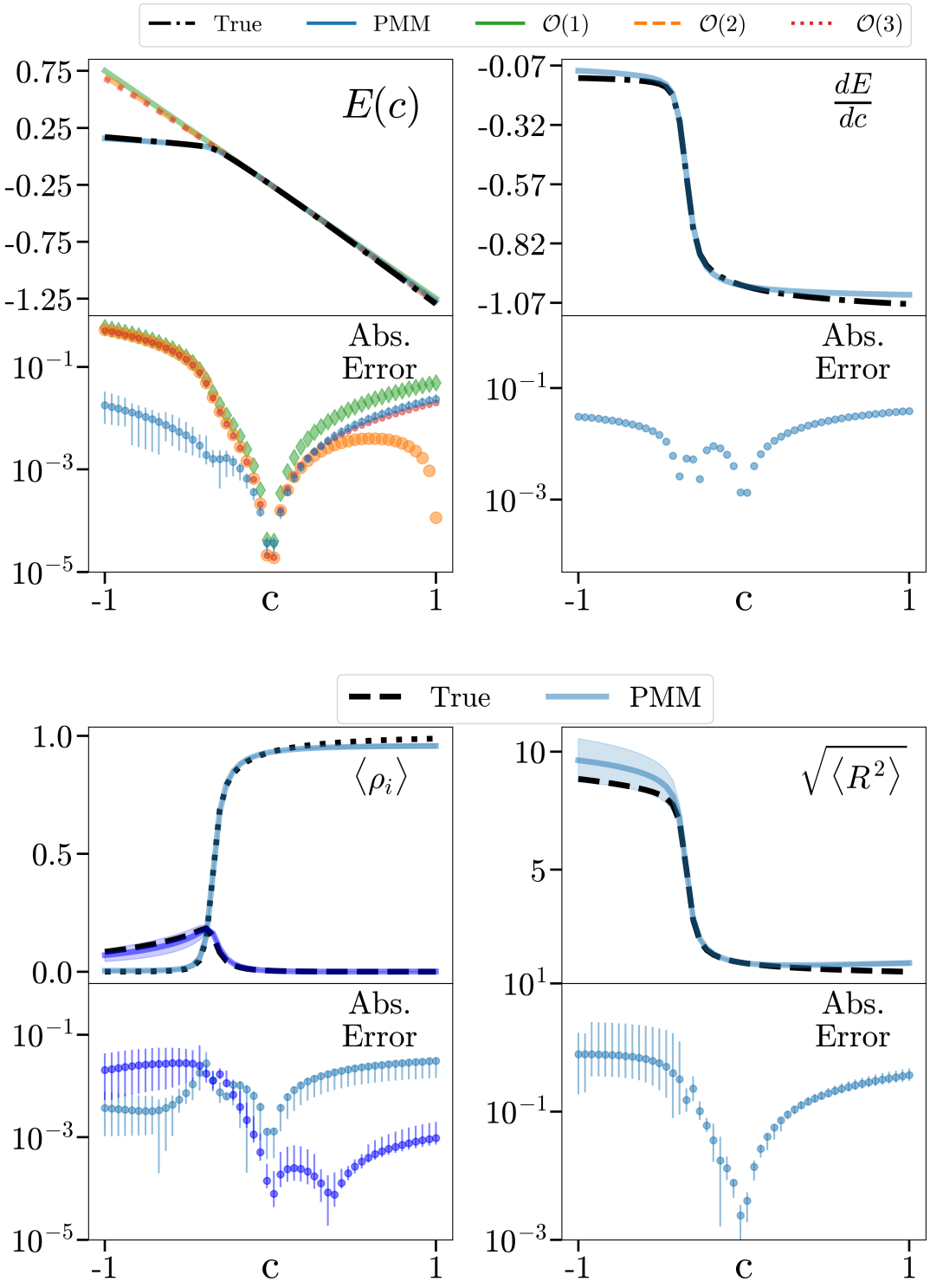

Figure 6.11 Ensemble of 9×9 PMM predictions— from left to right— for the ground-state energy, its derivative dE/dc, densities near and far from the origin, and charge radii. $\mathcal{O}(1)$, $\mathcal{O}(2)$, and $\mathcal{O}(3)$ indicates first-, second-, third-order corrections.

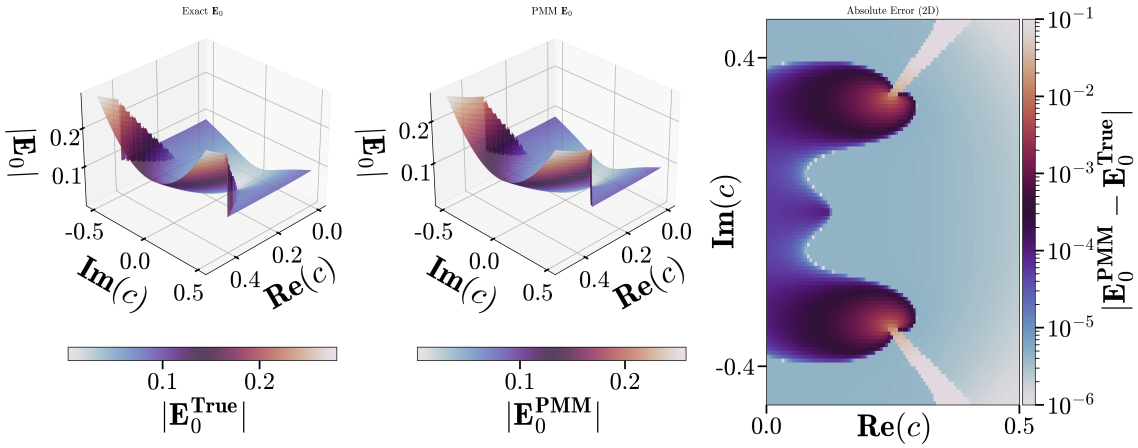

Figure 6.12 Extrapolation of a 9×9 PMM in the complex c-plane for H_0 without a bound state. Shown are the true values (left), PMM predicitions (middle), and absolute error (right).

6.4.2.2 Monte Carlo Lattice EFT

Building on the successes obtained with synthetic lattice data, we next integrate PMMs with real Quantum Monte Carlo (QMC) simulations. Here we present preliminary results where a PMM is trained on a Hamiltonian of the form

$$H = H_0 + H_I, \quad H_0 = K + (1-c)V_{2N} + V_{3N}, \quad H_I = cV_{2N}. \tag{6.43}$$

Although this Hamiltonian is not a full *ab initio* calculation, it employs many of the same techniques used in Monte Carlo Lattice EFT (MCLEFT) while focusing on a simplified, ^{6}Li-like system. The goal here is to benchmark PMMs against noisy, real QMC data, marking a step beyond purely synthetic simulations.

As shown in Fig. 6.13, the PMM trained only on first-order perturbative data successfully reconstructs the nonperturbative ground-state energy. As the model dimension increases, the PMM predictions converge toward the exact ground state, achieving accuracy within the error bounds of the QMC reference values. In comparison, first-order perturbation theory exhibits a deviation of approximately 3 MeV. While still preliminary, these results demonstrate that PMMs can infer nonperturbative physics directly from limited perturbative data. Ongoing work will explore this

same toy model further, focusing on extending the analysis to additional observables, such as density and charge radii, analogous to the earlier synthetic lattice study and identifying any fringe cases.

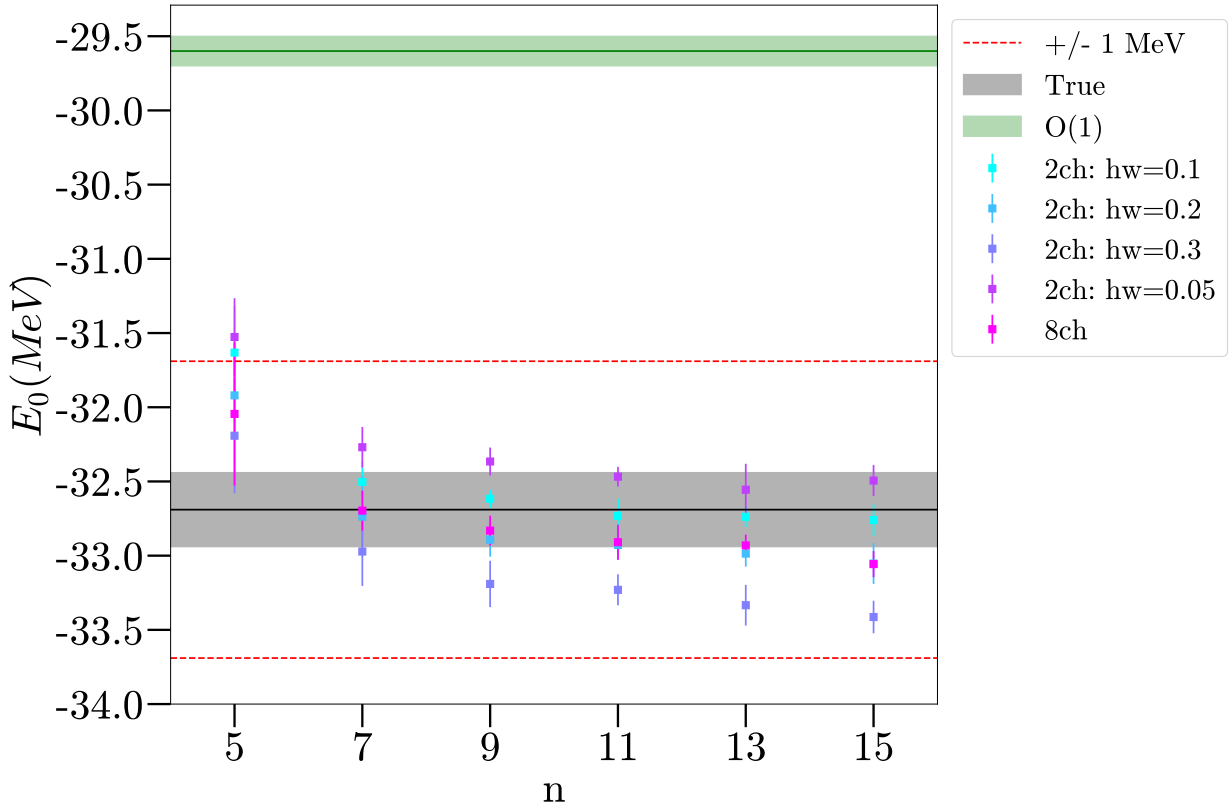

Figure 6.13 Ground-state energy of the ^{6}Li-like Hamiltonian. Green band: first-order perturbation from lattice MC. Gray band: true ground state. Blue–purple squares: PMM predictions for different initial-state lattice MC data. Pink squares: PMM trained on all initial states. Dashed red lines: ±1 MeV from the true ground state. x-axis shows the size of the PMM

6.4.3 Summary

In summary, the PMM framework provides a principled and versatile approach for enhancing lattice EFT by bridging physics-based modeling with data-driven learning. Rather than relying solely on explicit sampling or perturbative expansions, PMMs learn effective Hamiltonians and observables directly from Euclidean-time data. This allows them to reconstruct ground-state properties and capture nonperturbative effects that are inaccessible to conventional first-order

methods. Furthermore, their differentiable structure naturally supports analytic continuation into the complex plane, enabling the identification of singularities, branch points, and phase transitions.

Beyond their immediate application in lattice EFT, the PMM methodology is framework-agnostic and broadly applicable to other Monte Carlo–based approaches, such as lattice QCD and neural quantum state (NQS) methods, underscoring the broader significance of PMMs as a unifying tool for data-driven quantum many-body modeling. Training PMMs on computationally expensive QMC data opens the door to extrapolating into regimes that are currently intractable to conventional methods, which typically rely on first-order perturbation theory and suffer from the sign problem. In doing so, PMMs offer a systematic means of assessing higher-order corrections and circumventing the need increasingly prohibitive procedure for larger nuclei. In the long term, this framework could extend *ab initio* nuclear calculations to heavier systems and enable more accurate and interpretable predictions of nuclear observables.

6.5 Trotter Error Mitigation

This section explores the role of the Parametric Matrix Model (PMM) in mitigating Trotter errors in quantum computing, as introduced in Chapter 3. Quantum computing holds the promise of revolutionizing the simulation of complex nuclear systems, enabling computations that exceed the capabilities of classical supercomputers. However, we currently reside in the NISQ (Noisy Intermediate-Scale Quantum) era, where fully fault-tolerant quantum computers remain a future goal. In the meantime, error mitigation techniques, strategies that post-process noisy data to recover estimates closer to the true noiseless results, are critical for obtaining meaningful quantum computing outputs. In Chapter 3, various error mitigation strategies were discussed; here we focus specifically on mitigating Trotter errors.

A central challenge in simulating time evolution on quantum devices arises from the need to decompose the Hamiltonian operator, $H = \sum_l H_l$, which determines the energy of a quantum system. In practice, the time evolution operator over a small time step dt is often expressed as a product of individual operators.

$$\prod_l \exp(-iH_l dt) \tag{6.44}$$

This sequential decomposition is known as the Trotter approximation [92]. Higher-order corrections to this approximation, which systematically reduce the approximation error, are encapsulated in the Trotter-Suzuki formulae [86, 87, 15]. It is often convenient to define an effective Hamiltonian H_{eff} such that its time evolution operator matches the Trotter approximation for $U(dt)$.

$$\exp(-iH_{eff}dt) \approx \prod_{l} \exp(-iH_l dt) \tag{6.45}$$

Using the Trotter approximation, the quantum computer can then find the eigenvalues and eigenvectors of H_{eff}. In the limit $dt \to 0$, H_{eff} converges to the true Hamiltonian H. However, the number of quantum gate operations scales inversely with the magnitude of dt, and so a major challenge for current and near-term quantum computing is extrapolating observables to $dt = 0$ from values of dt that large in magnitude.

The PMM framework offers a promising pathway to mitigate these Trotter errors by learning effective representations that account for the discrepancy between the Trotter approximation and the ideal time evolution. In the coming sections, we detail how PMMs can be deployed to correct for these errors, effectively extrapolating observables to the $dt = 0$ limit from data obtained at finite time steps. This novel approach, by embedding the structure of the governing equations into a parametric model, provides an alternative method for error mitigation that is both systematic and adaptable.

6.5.1 Theory

Consider a Hamiltonian composed of l local, generally non-commuting components:

$$H = \sum_{i}^{l} H_i \tag{6.46}$$

Because circuit-based quantum computers require unitary operations, we approximate the full time evolution operator by applying the Trotter decomposition. Specifically, the Trotter approximation expresses the time evolution operator for a small time step dt as

$$e^{-iHdt} \approx \prod_{i}^{l} e^{-iH_l dt} + O(dt^2) \tag{6.47}$$

In the limit as $dt \to 0$, the Trotter error (which scales as $O(dt^2)$ vanishes. However, smaller dt means that more unitary gates must be applied, which—if the quantum circuit becomes too deep—can lead to quantum decoherence and loss of useful information. In practice, one is forced to work with relatively large dt values (typically around $dt \leq \pi/||H||_2^2$) and then extrapolate the observables to the $dt = 0$ limit. Once a Hamiltonian is implemented on a quantum computer, several efficient algorithms extract energy levels by analyzing the complex phases acquired during time evolution of quantum states [46, 1, 28, 88, 18, 77, 71, 5, 21].

Since the underlying equations here are similar to those presented in the previous sections, we assume that the PMM employs an analogous ansätze. In this framework, we mirror the physical dynamics by introducing learnable matrix parameters. We parameterize the effective Hamiltonian M as a sum of l local learnable matrices

$$M = \sum_{i}^{l} \underline{M_i} \tag{6.48}$$

Where $M \in \mathbb{C}^{n \times n}$ and $n << 2^N$ (with N being the number of qubits in the full Hamiltonian). We then apply the Trotter approximation to the PMM Hamiltonian:

$$e^{(-iMdt)} = \prod_{i}^{l} e^{-i\underline{M}_l} \tag{6.49}$$

Because the reduced dimension n is very small, we can easily diagonalize the effective time evolution operator classically. Denote

$$U_M(dt) = e^{-iMdt} \overset{EVD}{\to} V\Lambda V^{\dagger} \tag{6.50}$$

so that the eigenvalues are given by

$$\begin{aligned} e^{-i\epsilon_k dt} &= \lambda \\ \epsilon_k &= \left[\frac{ln(\lambda)}{-idt}\right]_k \end{aligned} \tag{6.51}$$

The learnable matrices M_i of the PMM are optimized by minimizing the mean squared error between these effective PMM energies ϵ_k and the true energy values E_k (obtained, for instance,

via phase estimation on the quantum computer). In other words, we decompose the effective Hamiltonian in analogy with the physical Hamiltonian, and train the PMM so that its eigenvalues reproduce the target energy spectrum.

Once the PMM is trained, extrapolation to the $dt = 0$ regime is achieved simply by rewriting the effective Hamiltonian as in equation (6.48) and performing a full diagonalization. This yields the energy levels of interest in the ideal, noiseless limit.

6.5.2 Results

In this section we present numerical results demonstrating the efficacy of the PMM framework for mitigating Trotter errors in quantum simulations. Our benchmark systems are quantum spin models based on a one-dimensional Heisenberg Hamiltonian, which can include either only nearest to nearest neighbor interactions or an antisymmetric spin-exchange term known as the Dzyaloshinskii–Moriya (DM) ref[]. Although we focus primarily on the DM interaction, we briefly highlight results from the standard Heisenberg model as further evidence of the PMM's versatility.

6.5.2.1 Heisenberg model

We first consider a Heisenberg spin model for $N = 10$ qubits, which has garnered significant interest in studies of quantum coherence and entanglement. The Hamiltonian under periodic boundary conditions is given by

$$H = B \sum_{i}^{N} r_i \sigma_i^z + J \sum_{i}^{N} (\sigma_i^z \sigma_{i+1}^z + \sigma_i^x \sigma_{i+1}^x + \sigma_i^y \sigma_{i+1}^y) + D \sum_{i}^{N} (\sigma_i^x \sigma_{i+1}^y - \sigma_i^y \sigma_{i+1}^x) \tag{6.52}$$

where B is an external magnetic field, J is the exchange interaction strength, and D characterizes the DM interaction. The Trotter approximation for this Hamiltonian naturally splits it into five terms: the field term and four interaction terms (the symmetric nearest-neighbor and antisymmetric DM interactions further partitioned by parity). For this system, classical simulations are feasible; we generate data for time steps dt starting at $dt \leq \pi/||H||_2^2$.

To convert the problem from an extrapolation to an interpolation task, we calculate the eigenvalues for both positive and negative dt values. Figure 6.14 plots the lowest three energy eigenvalues of the effective Hamiltonian H_{eff} versus the time step dt. We compare the PMM predictions with

those obtained using a Multilayer Perceptron (MLP) s well as with standard polynomial interpolation, further detail ref [23]. Notably, all training and validation samples are located away from dt to reflect realistic conditions on current quantum hardware.

The inset in Figure 6.14 displays the relative errors at $dt = 0$ for the predicted energies. Here, the PMM outperforms both the MLP and polynomial interpolation methods across all low-lying eigenvalues, achieving an error reduction of more than an order of magnitude for the ground state energy E_0.

Figures 6.15 and 6.16 provides similar comparisons for an $N = 9$ qubit system. Figure 6.15 shows results for a Heisenberg model with only nearest-neighbor interactions, while panel Figure 6.16 illustrates the case with the DM term. In particular, panel Figure 6.15 highlights the intricate behavior of eigenvalues near zero, where their mutual interactions are most pronounced. Here, the PMM is able to navigate these complex interactions and accurately predict the $dt = 0$ results from training data acquired at larger $dt = 0$ values, whereas the polynomial interpolation method fails to capture these subtleties.

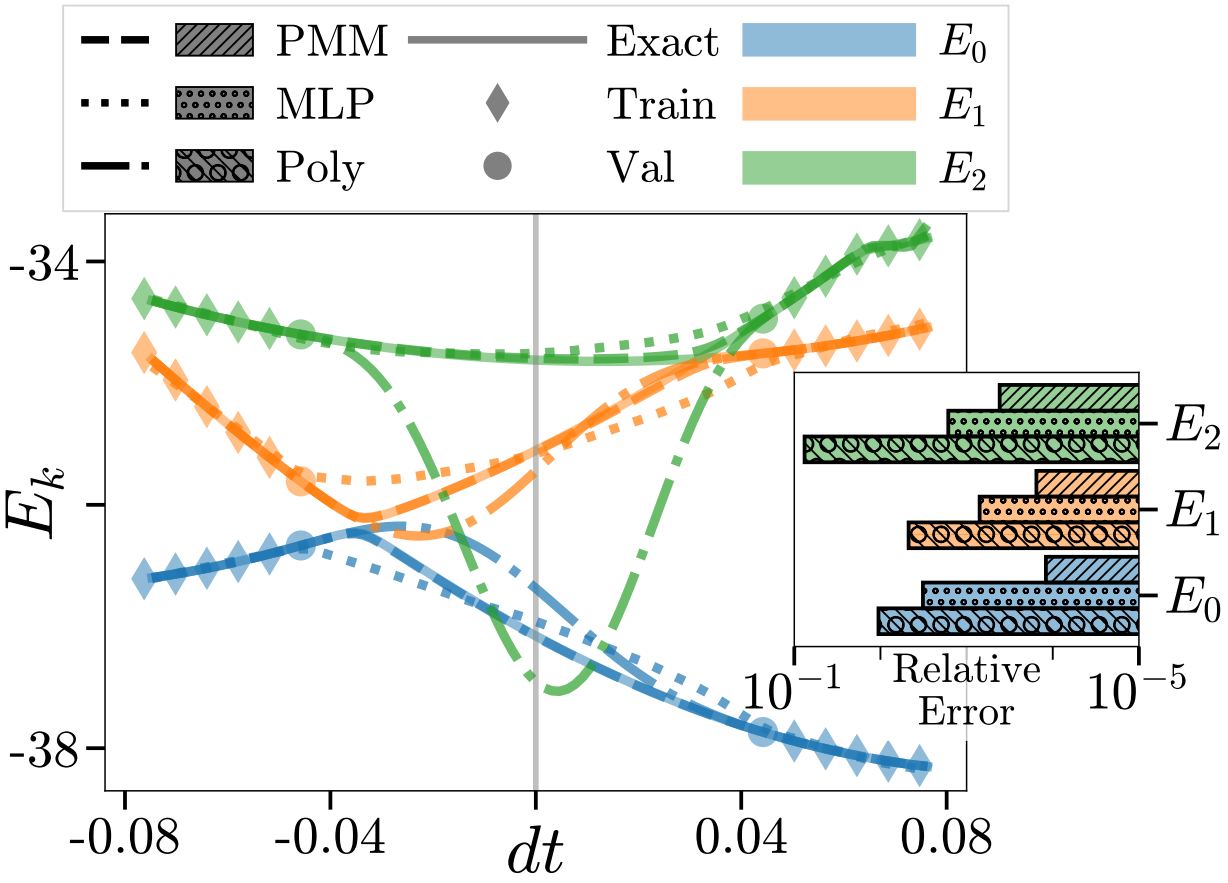

Figure 6.14 Extrapolated Trotter approximation for quantum computing simulations. We plot the lowest three energies of the effective Hamiltonian for the one-dimensional Heisenberg model with DM interactions versus time step dt. We compare results obtained using a PMM (dashed), Multilayer Perceptron (MLP, dotted), and polynomial interpolation (Poly, dash-dotted). All training (diamonds) and validation (circles) samples are located away from $dt = 0$, where data acquisition on a quantum computer would be practical. The inset shows the relative error in the predicted energies at $dt = 0$ for the three models. Figure taken from [23]

6.5.3 Summary

In summary, our results demonstrate that the PMM framework provides a powerful method for mitigating Trotter errors in quantum simulations. By mapping one eigenvalue problem into another through a learnable reduced model, the PMM leverages the intrinsic properties of eigenvalues—such as avoided level crossings (i.e eigenvalue repulsion)—to accurately predict system behavior at $dt = 0$, even when the training data is obtained from much larger time steps. This built-

in capacity to capture the complex interactions between eigenvalues allows the PMM to outperform traditional approaches like multilayer perceptrons and polynomial interpolation, delivering significantly lower errors.Consequently, the PMM's ability to translate the eigenvalue structure of the original Hamiltonian into an effective reduced eigenvalue problem is key to its superior performance in extrapolating to the noiseless limit in current quantum computing applications.

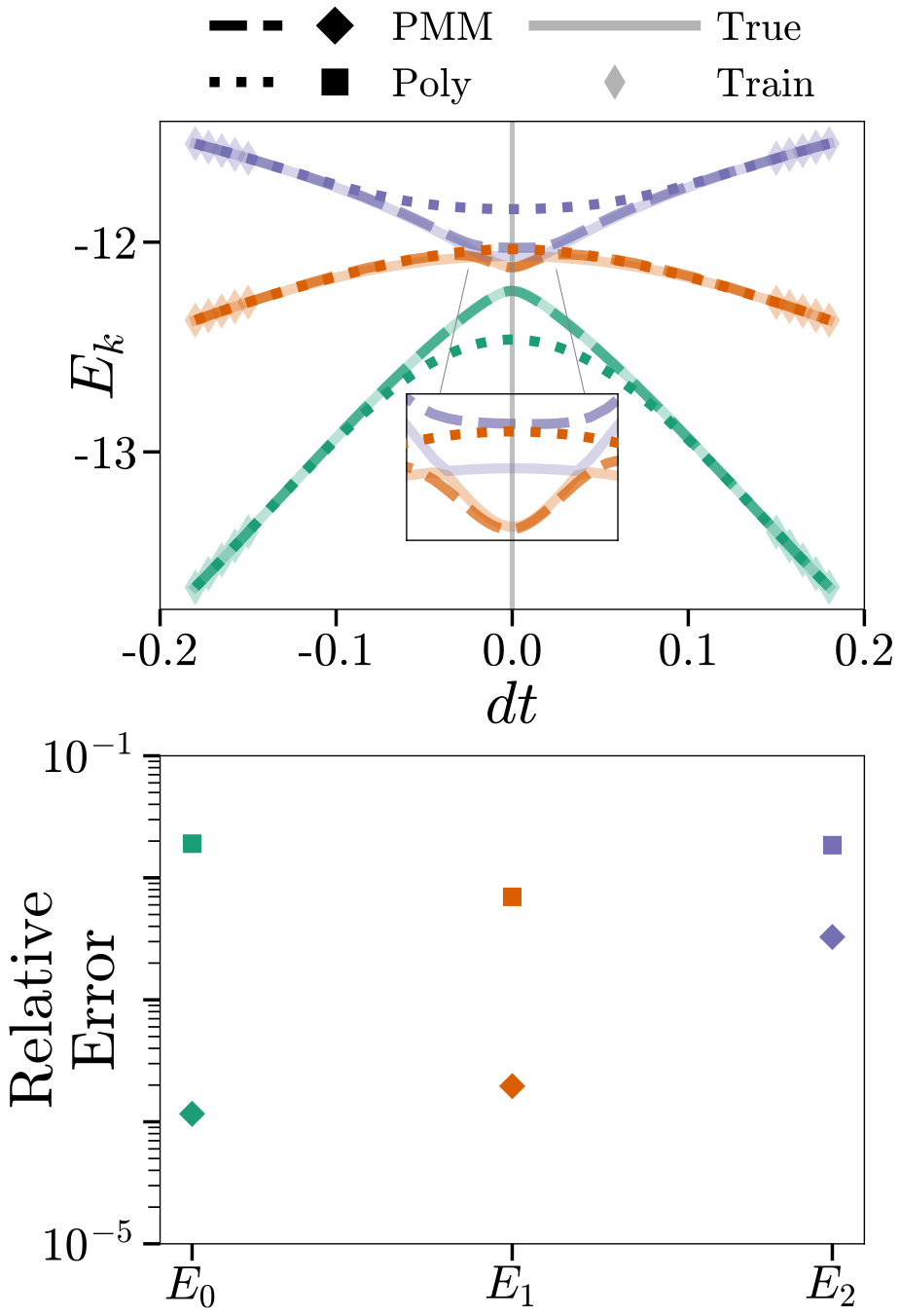

Figure 6.15 Results for selected energy levels in the one-dimensional Heisenberg spin chain with nearest-neighbor and next-nearest-neighbor interactions (top) as well as the relative error for the extrapolated values at $dt = 0$ (bottom). We show the 10×10 PMM results (dashed lines, square diamonds), polynomial interpolation results (dotted lines, squares), exact diagonalization calculation of Trotter groundtruth (solid lines), and Trotter training data (thin diamonds). The vertical grey line highlights $dt = 0$.

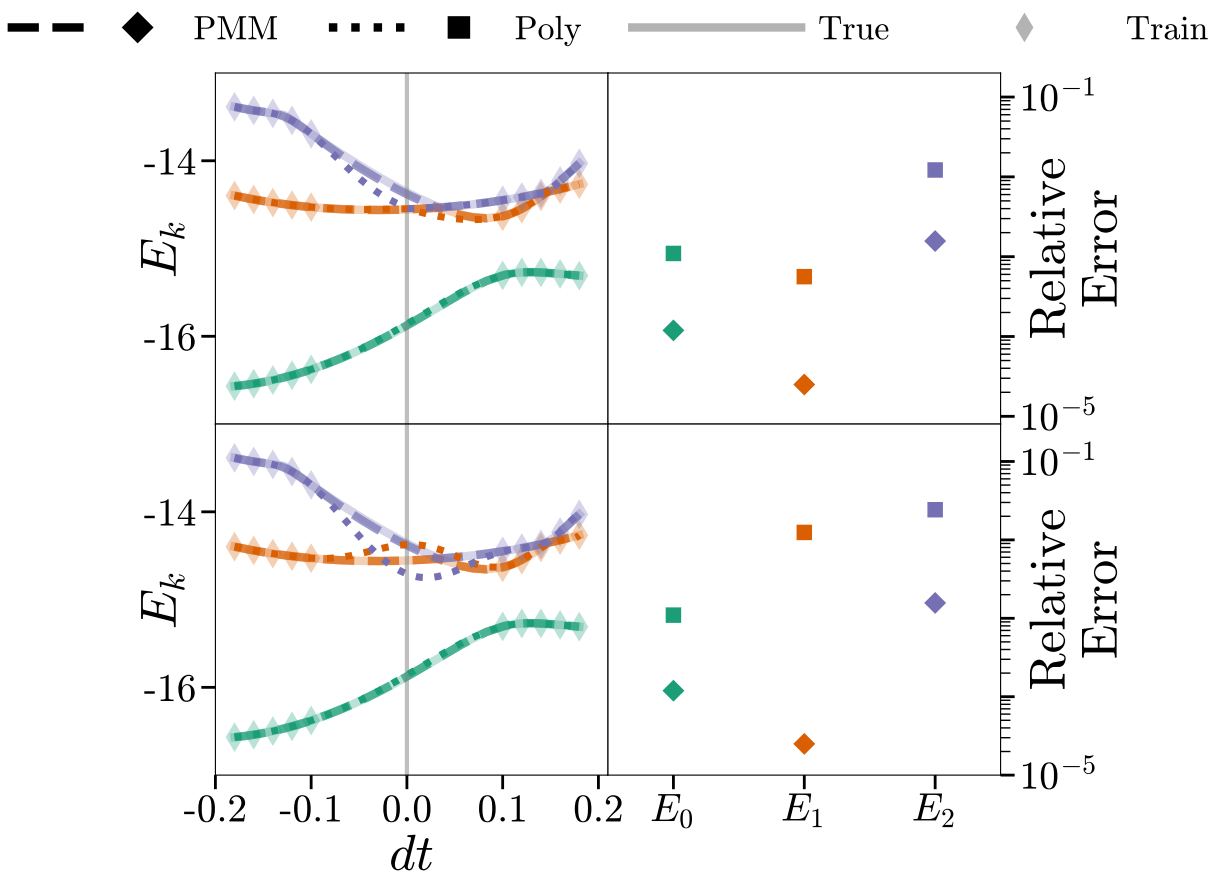

Figure 6.16 Results for selected energy levels in the one-dimensional Heisenberg spin chain with nearestneighbor and DM interaction. The first row shows the results when polynomial interpolation was allowed to treat the level crossings as true level crossings. By contrast, in the second row the polynomial received the data in its original form. We show the 5×5 PMM (dashed lines, square diamonds), polynomial interpolation results (dotted lines, squares), exact diagonalization calculation of Trotter ground-truth (solid lines), and Trotter training data (thin diamonds). The vertical gray line highlights $dt = 0$.

6.6 Computational Fluid Dynamics

Up to now, our Parametric Matrix Model (PMM) framework has been defined by deriving reduced, parameterized operators via Proper Orthogonal Decomposition (POD) so as to mimic known governing equations. In each of the previous examples—Hamiltonian simulation, Euclidean time evolution, and Trotter error mitigation—we saw that performing a POD construction naturally reveals the correct PMM ansätze in the lower-dimensional latent space. Although this

POD-based extension originally emerged from our work on the nonlinear Gross–Pitaevskii equation for Bose–Einstein condensates [24, 22], and has thus far been applied primarily within nuclear physics, the approach is in fact completely general: any system governed by known equations can be emulated by a PMM.

In this spirit, we now turn to one of the classical applications of POD, computational fluid dynamics, and demonstrate how PMMs can efficiently emulate the fully nonlinear Navier–Stokes equations. While modern fluid-flow surrogates are often built using Physics-Informed Neural Networks (PINNs) or Sparse Identification of Nonlinear Dynamics (SINDy). While SINDy infers sparse symbolic models directly from data, PINNs enforce governing equations through penalty terms in the loss function. Despite their successes, both approaches can be sensitive to hyperparameter tuning, overfitting, and noise. PINNs in particular may fail to preserve exact conservation laws or symmetries and are often overparameterized, making them slower and less accurate than simply running a lower-fidelity version of the original high-fidelity solver [34, 58]. By contrast, our PMM inherits the structure of a high-fidelity fluid solver, complete with operator splitting and projection steps, directly in its reduced form. This ensures that fundamental constraints (e.g., mass conservation and incompressibility) are built into the model by construction, yielding interpretability, strict adherence to the governing physics, and computational efficiency far beyond that of generic neural-network architectures.

In the sections that follow, we detail how we project the incompressible Navier–Stokes equations into a latent POD basis and replace each operator with a small, trainable matrix. Nonlinear terms are handled via precomputed (or learnable) tensors that avoid repeated high-dimensional projections. The resulting PMM not only reproduces the essential fluid dynamics but does so with a fraction of the computational cost, opening the door to real-time emulation and uncertainty quantification for complex flows.

6.6.1 Navier-Stokes

The Navier-Stokes (NS) equations describe the motion of fluid substances and are foundational in fluid dynamics. These equations govern the conservation of momentum and mass in a fluid,

accounting for viscous forces, pressure gradients, and external forces. For incompressible flows, a key simplification arises: the fluid density remains constant, leading to a divergence-free velocity field.

$$\begin{aligned} \frac{\partial u}{\partial t} + (u \cdot \nabla)u &= -\frac{1}{\rho}\nabla p + \nu \nabla^2 u + f \\ \nabla \cdot u &= 0 \end{aligned} \tag{6.53}$$

where u is the velocity field, p is the pressure, ρ is the density, ν is the kinematic viscosity, and f represents external forces.

The incompressible NS equations are expressed as a system of partial differential equations, comprising the momentum equation and the incompressibility condition. These equations model a wide range of phenomena, from airflows around wings to ocean currents, and serve as a cornerstone for both theoretical investigations and practical simulations in fluid mechanics. Solving the incompressible NS equations is computationally intensive, as the coupled equations necessitate sophisticated numerical methods to enforce the divergence-free constraint while accurately capturing complex fluid behaviors.

To address the challenges of solving the incompressible Navier-Stokes equations, computational methods like the Stable Fluids algorithm have been developed. Introduced by Jos Stam [84]. It combines operator splitting with implicit methods to ensure numerical stability. A key feature is a pressure projection step that enforces the incompressibility condition by solving a Poisson equation for the pressure field, ensuring a divergence-free velocity field at each timestep. An algorithmic description can be found in methods.

In our study, we adopt the Stable Fluids algorithm as the governing framework but shift the focus from its implicit methods to parametrizing its functional form. To achieve a reduced-order representation of the algorithm, we apply Proper Orthogonal Decomposition (POD) as described in Section 6.2. The POD process results in a reduced set of equations, where unknown operators are parameterized to construct our Parametric Matrix Model (PMM).

A particular challenge in applying POD to the Navier-Stokes equations arises from their inherent nonlinearity. For instance, the convective (or self-advection) term involves a Hadamard (element-

wise) product, which, in its full form, requires projecting latent-space variables back to the full space, applying the nonlinear operation, and then re-projecting onto the latent space. Such operations are computationally prohibitive, especially within a machine learning context where repeated interaction with the full-dimensional space during gradient descent must be minimized.

To overcome this, we reformulate the nonlinear term by grouping the projection operators and precomputing a rank-3 tensor that encapsulates the necessary contractions. For example, the nonlinear term in the latent space

$$v^x \odot g^x v^x = P^{xT}_{qi} P^x_{ir} v^x_r P^x_{il} g^x_{ld} v^x_d \tag{6.54}$$

is re-expressed as

$$v^x \odot g^x v^x = \mathbb{T}^x_{q|rl} v^x_r g^x_{ld} v^x_d \tag{6.55}$$

Where $\mathbb{T}^x_{q|rl}$ functions as a hypernetwork, contracting multiple shared indices and confining its physical dimensions to the latent space. In cases where the nonlinear interactions exceed second order and precomputation becomes inefficient, we allow $\mathbb{T}^x_{q|rl}$ o become a learnable tensor, denoted by $\underline{\mathbb{T}}_{q|rl}$. In practice, even when precomputation is feasible, learning this tensor endows the PMM with enhanced flexibility and often permits a reduction in the number of principal components required.

Thus, for the incompressible Navier-Stokes equations, the PMM ansätze becomes

$$\begin{aligned} \frac{\partial u}{\partial t} + \underline{\mathbb{T}}_{q|rl} u_r \underline{\nabla}_{ld} u_d &= -\frac{1}{\rho} \underline{\nabla} p + \nu \underline{\nabla}^2 u + f \\ \underline{\nabla} \cdot u &= 0 \end{aligned} \tag{6.56}$$

Here, if the external forcing (or inflow conditions) is not known a priori, it can also be treated as a learnable component. Notably, since the PMM is designed to solve the Navier-Stokes equations, the underlying physical constraints are inherently embedded in the model. As a result, the PMM output is always a solution of the NS equations, even if it might deviate slightly from the true solution due to model limitations. Further details of the POD-PMM construction can be found in the appendix .

6.6.2 Results

In this study, we evaluated the PMM on a canonical incompressible flow problem: the flow over a circular cylinder, which generates a characteristic von Kármán vortex street. This flow pattern, shown in Figure 6.17, emerges when fluid passes an obstacle and alternately sheds vortices downstream, forming a repeating pattern of swirling eddies. The phenomenon is a classic example of fluid instability and nonlinear dynamics, appearing in natural systems like clouds downstream of islands in the atmosphere. The "ground truth" snapshots used for training and validation were generated using the lattice Boltzmann method (LBM), adapted from [48], a mesoscale computational fluid dynamics approach that models fluid motion by simulating the statistical evolution of discrete particle populations on a lattice. Rather than solving the Navier–Stokes equations directly, LBM recovers them as an emergent behavior in the hydrodynamic limit, offering excellent parallel scalability and numerical stability for complex boundary conditions and geometries. Unlike our PMM, which is based on an explicit "Stable Fluids" time-stepping scheme, LBM effectively incorporates implicit dynamics through its collision–streaming formulation, making the PMM's time-step parameter dt an important hyperparameter controlling stability and accuracy during training.

Although we did not perform a side-by-side comparison with other reduced-order modeling techniques in this work, earlier studies applying Physics-Informed Neural Networks (PINNs) to the same flow reported significant difficulties in capturing its strongly nonlinear, vortex-shedding behavior [19, 42]. These works also found that the resulting PINN solutions contained unphysical artifacts, such as discontinuities or non-smooth transitions, preventing a coherent evolution from the initial steady state to the periodic vortex-shedding regime. Furthermore, these PINNs were heavily overparameterized, often requiring millions of trainable weights, leading to prohibitively slow convergence and high computational cost. In contrast in Figure 6.17, our PMM employed only 5×5 learnable matrices, yet achieved a faithful and physically consistent temporal evolution. Even when the PMM is not exact, each prediction remains a valid solution of the Navier–Stokes equations by construction, producing interpretable and dynamically coherent flow trajectories from steady-state onset to the fully developed von Kármán vortex street.

Future work will focus on extending this proof-of-concept study along two main directions. First, we aim to investigate whether the PMM can learn parametric dependencies beyond time evolution, such as the Reynolds number. By training on flow fields spanning a range of Reynolds numbers, we can test the model's ability to interpolate and extrapolate across regimes—transitioning, for example, from laminar to turbulent flow behavior—thereby exploring the PMM's potential as a compact surrogate for parameterized dynamical systems. Second, we plan to move beyond the explicit Euler time-stepping used in the current implementation and incorporate differentiable implicit solvers, such as those available in the JAX `diffrax` library. This would enable more stable and accurate integration, potentially improving long-term predictive performance and extending the applicability of PMMs to more complex fluid and multiphysics problems.

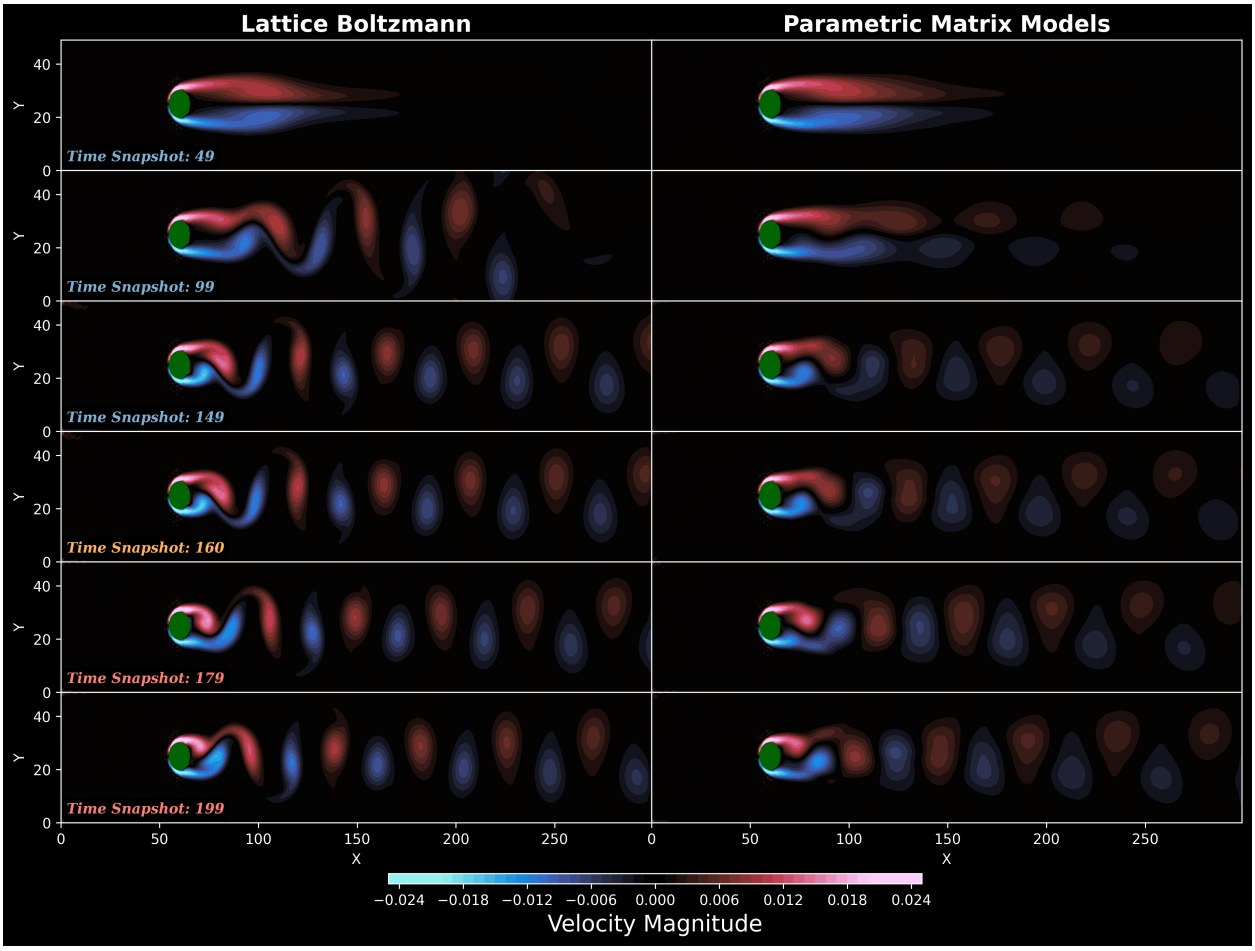

Figure 6.17 Snapshots of a Kármán vortex street (flow over a cylinder) at different times. Left: true lattice Boltzmann results. Right: PMM predictions. Blue snapshots indicate training times, orange snapshots indicate validation times, and red snapshots indicate test times.

6.6.3 Summary

In summary, the reduced-order modeling of the incompressible Navier-Stokes equations via the PMM framework offers a promising and efficient strategy for complex fluid dynamics simulations. By parametrizing the governing equations through Proper Orthogonal Decomposition, the PMM incorporates the essential physical constraints of fluid flow while significantly reducing computational demands. The key innovation lies in addressing the nonlinearity, specifically, the self-advection term, by precomputing or learning an effective rank-3 tensor that encapsulates the necessary projections within the latent space. This approach circumvents the high cost of repeated

full-space operations during gradient descent, ensuring computational efficiency. Consequently, the PMM is capable of providing robust, interpretable solutions that remain faithful to the underlying Navier-Stokes dynamics, even when the model is supplied with limited data.

6.7 General PMM

Thus far, we have illustrated how Parametric Matrix Models (PMMs) can faithfully emulate complex physical systems, from quantum eigenvalue problems and Euclidean time evolution to Trotter error mitigation and nonlinear fluid dynamics, by embedding known governing equations directly into a reduced, trainable operator framework. In this final section, we venture beyond the world of physics into the broader landscape of machine learning, exploring whether PMMs can serve as a general-purpose function approximator for regression and classification tasks.

Today's machine-learning toolkit is dominated by deep neural networks, celebrated for their universal approximation capabilities and deployed in domains ranging from financial forecasting to computer vision, stable-diffusion image generation, and large language models like ChatGPT. Yet the tremendous success of these models comes at a steep price: modern architectures often require hundreds of millions—or even billions—of parameters, leading to substantial computational and energy costs during training and inference. Thus this singular focus on neural architectures means that relatively little attention has been paid to alternative approaches that lie on a different axis in machine learning, despite their potential to offer greater interpretability and efficiency.

Here, we ask whether a PMM architecture, grounded not in differential equations but in the quantum-mechanics observable formalism, can offer comparable predictive accuracy with dramatically fewer parameters. By representing the target function implicitly through small, input-dependent matrix equations, we open a new axis in the space of machine-learning architectures: one that trades depth and width for analyticity and interpretability.

In this section we will define a PMM ansätze inspired by observables in quantum mechanics, parameterized so as to capture nonlinear regression and classification boundaries. Benchmark its performance on standard regression and classification datasets, directly comparing accuracy and parameter count against multilayer perceptrons and other conventional models. Demonstrate param-

eter efficiency, showing that PMMs can achieve similar—or in some cases better—generalization with orders of magnitude fewer trainable weights

While we do not yet extend PMMs to tasks such as image synthesis or language modeling, the results presented here will lay the groundwork for future exploration of PMMs as a lightweight, interpretable alternative in domains traditionally dominated by deep neural networks.

6.7.1 Theory

We address the general problem of regression and classification in machine learning where the model learns to reproduce feature-label pairs $\{(x_i,\ y_i) : x_i \in \mathbb{R}^p,\ y_i \in \mathbb{R}^q\}$ and generalize to unseen data. Regression and classification are fundamentally the same problem, differentiated only by the nature of the labels $\{y_i\}$ and therefore suitable choices of loss functions.

The simplest PMM for this task uses a primary matrix which is affine in the input features, as shown previously in Eq.(6.13). This is the form we have chosen for our regression experiments as well as our hybrid transfer learning experiments. The form of the primary matrix or matrices can be modified to accommodate known properties of the data, for example the image classification PMM discussed [23].

The first $\overline{r}$ eigenvectors associated with the largest magnitude eigenvalues form bilinears with Hermitian secondary matrices $\underline{\Delta_{kij}} \in^n$ where $1 \leq k \leq q$ and $1 \leq i, j \leq \overline{r}$ and $\underline{\Delta_{kij}} = \underline{\Delta_{kji}}$ to produce the output of the PMM. That is, for each output index k and each pair of eigenvector indices i, j there is an independent secondary matrix $\underline{\Delta_{kij}}$. These bilinears—which can be thought of as expectation values (transition amplitudes) when $i = j$ ($i \neq j$)—are summed together with a trainable bias vector $\underline{g} \in \mathbb{R}^q$ and a fixed bias proportional to the spectral norm of the secondary matrices to form the output of the PMM,

$$z_k = \underline{g_k} + \sum_{i,j=1}^{\overline{r}} \left| v^{(i)\dagger} \underline{\Delta}_{kij} v^{(j)} \right|^2 - \frac{1}{2} \left| \underline{\Delta}_{kij} \right|_2^2. \tag{6.57}$$

Equivalently, the sum may be restricted to $i \leq j$ for efficiency. The fixed bias term is a deliberate addition to ensure that the output of the PMM is both unbounded and invariant under suitable unitary transformations of the trainable matrices. This output vector may be augmented by fixed

or trainable activation functions, such as the softmax function in the case of classification.

The form of this PMM 6.18, which we call the affine observable PMM (AO-PMM) due to the analogy with observables and transition probabilities in quantum mechanics, is not motivated by any specific data, but instead purely by the desire to generalize the affine eigenvalue PMM to cases with multiple outputs of arbitrary algebraic order.

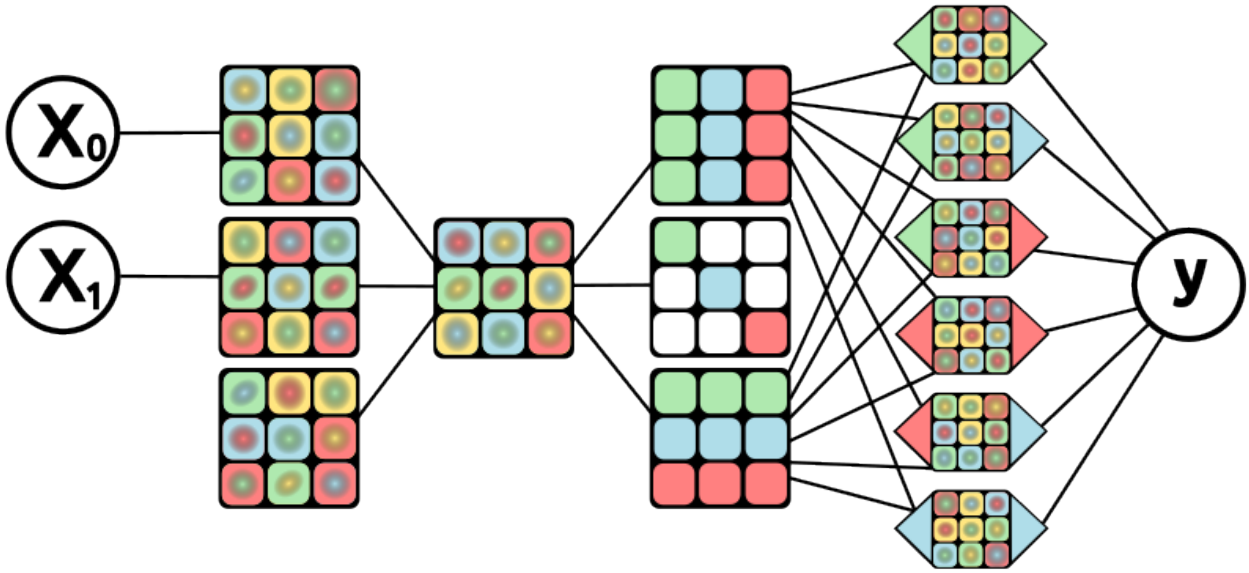

Figure 6.18 Diagrammatic representation of the affine observable PMM architecture.

6.7.2 Results

We now demonstrate the versatility of the Parametric Matrix Model (PMM) as a general-purpose universal approximator for machine learning tasks. Two representative applications are considered: first, interpolation and extrapolation in multivariate regression, and second, supervised image classification.

6.7.2.1 Regression

We first examine the performance of PMMs still within the context of scientific computing, focusing on multivariate regression tasks. Specifically, we compare the general affine PMM against several widely used machine learning techniques, encompassing both online and offline

methods—namely, Multilayer Perceptrons (MLPs), Extreme Gradient Boosting (XGB), and Kernel Ridge Regression (KRR), among others.

Our benchmark suite comprises thirteen analytic two-dimensional test functions, two additional families of synthetic functions, and two real-world datasets: the NASA airfoil self-noise dataset and the CERN dielectron collision dataset. For all methods other than the PMM, we performed an extensive grid search with hyperparameter tuning to ensure optimal performance.

Figure 6.19 presents the normalized mean absolute error (NMAE) on withheld test data across the seventeen regression benchmarks. Horizontal lines indicate the mean performance across benchmarks. The PMM consistently outperforms all other methods, achieving the lowest error in fifteen out of seventeen cases. In several examples, such as the Franke, Cliff, and Runge functions, the PMM attains NMAE values that are orders of magnitude smaller than those of competing methods. While also requiring up to an order of magnitude fewer trainable parameters then the MLP.

This improvement can be attributed to the smoother functional behavior of the PMM between training points. Whereas standard machine learning algorithms often exhibit oscillatory artifacts in regions with sparse data, the PMM naturally mitigates such behavior. This arises from the model's eigenvector-based representation: when approaching oscillatory regions, nearby eigenmodes can undergo avoided crossings, effectively suppressing spurious oscillations and enforcing smoother interpolations.

Further implementation details, including the test functions, parameter counts, hyperparameter configurations, and computational complexity are provided in Ref. [23].

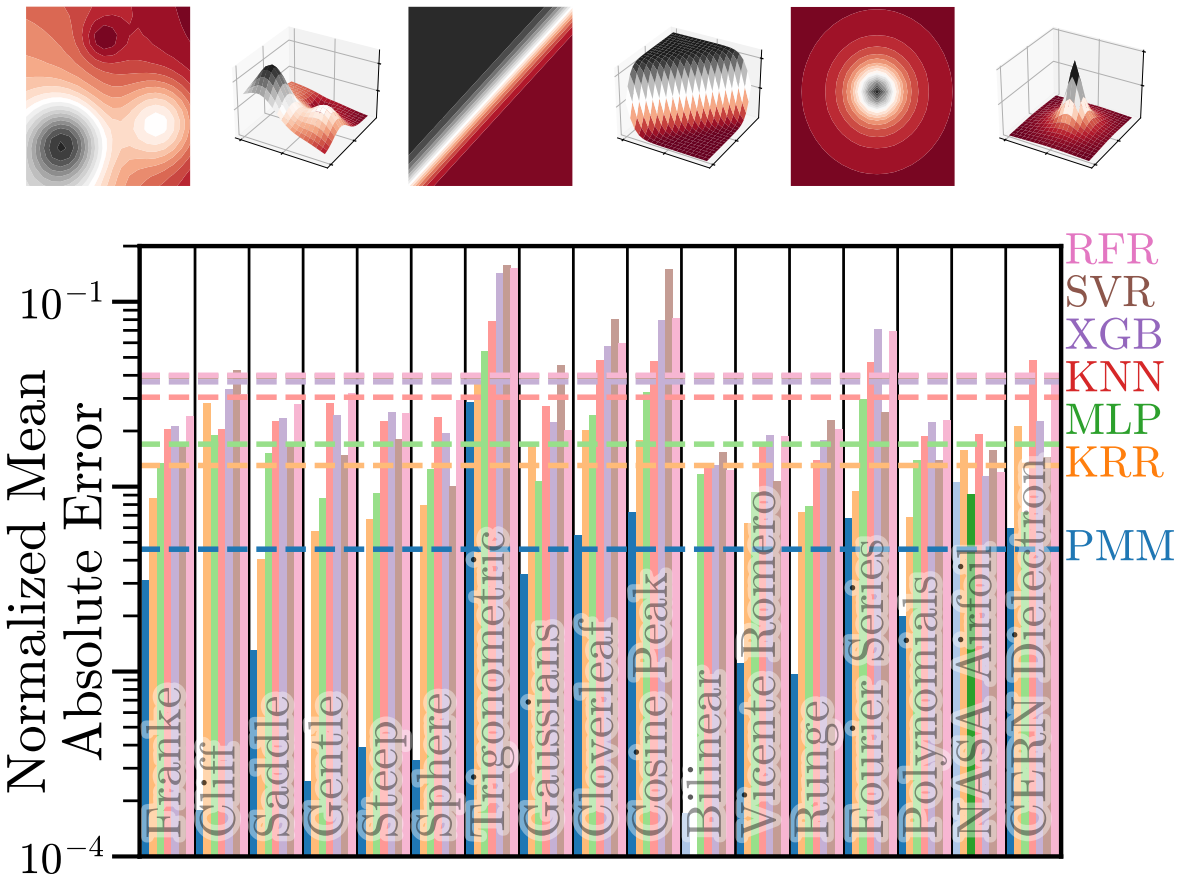

Figure 6.19 Top: 2/3D rendering of the Franke,Cliff, and Runge function. Bottem: Performance on regression problems. Normalized mean absolute error on withheld test data for the PMM (blue) compared against several standard techniques: Kernel Ridge Regression (KRR, orange), Multilayer Perceptron (MLP, green), k-Nearest Neighbors (KNN, red), Extreme Gradient Boosting (XGB, purple), Support Vector Regression (SVR, brown), and Random Forest Regression (RFR, pink). Normalized mean absolute error on provided training and validation data is shown for selected datasets as contrasting squares. Mean performance on withheld test data across all problems are shown as horizontal lines. Figure taken from [23]

6.7.2.2 Classification

Moving beyond physics-oriented and scientific computing tasks, we next apply PMMs to computer vision, specifically supervised image classification. Image classification involves training a model to correctly assign input images to predefined categories based on learned visual features—an essential benchmark for testing generalization and representational power in machine

learning architectures.

We evaluate both the standard affine PMM and a convolutional variant (ConvPMM) on three widely benchmarked grayscale datasets: MNIST handwritten digits [50], Fashion-MNIST clothing images [96], and EMNIST Balanced character images [20]. While these datasets are comparatively simple, they remain among the most extensively tested benchmarks in the field, providing a rigorous and standardized comparison across architectures. Our goal here is not to compete with the latest large-scale models, but to assess how effectively the PMM—originally developed for scientific data—can generalize to structured vision tasks.

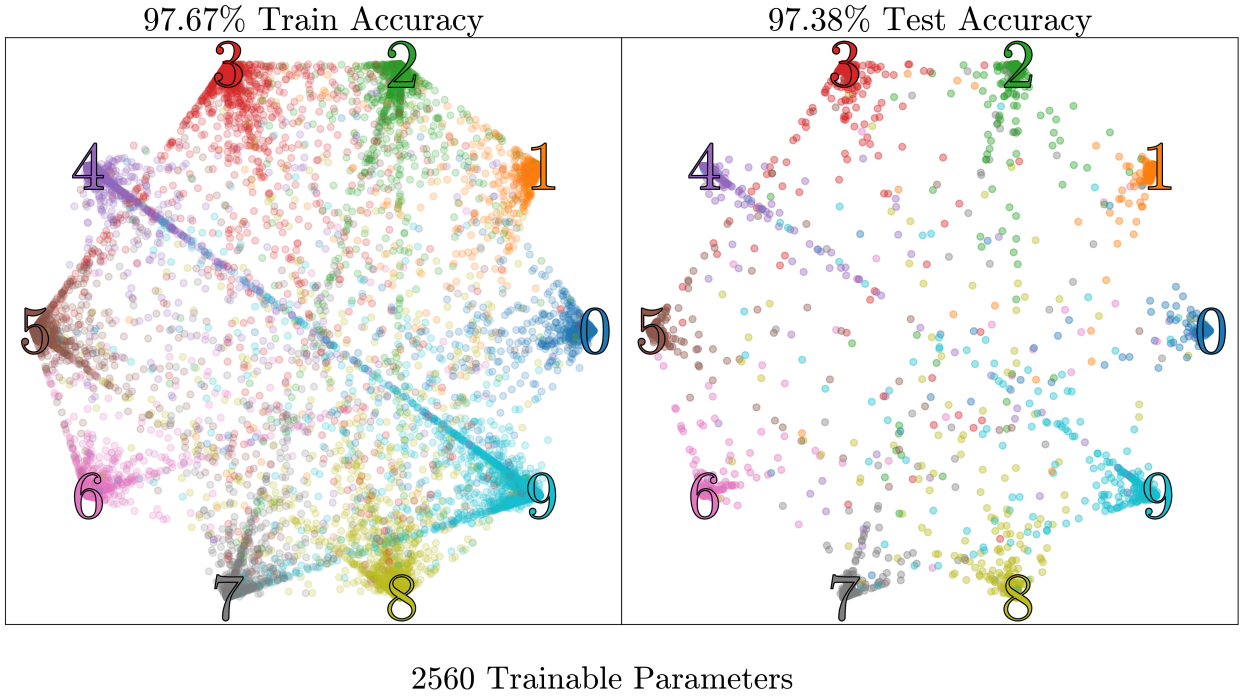

Figure 6.20 Visualization of a PMM trained on the MNIST dataset 6.1. Numbers are arranged on a circle, and PMM predictions for each digit are colored according to the corresponding number. Correlations between digits are indicated by lines connecting them, e.g., a line from 9 to 4 shows a relationship captured by the PMM. Left: training set predictions. Right: test set predictions [22].

Figure 6.1 summarizes the test-set accuracies and trainable parameter counts. Across all datasets, the PMM achieves competitive accuracy while using one to two orders of magnitude fewer parameters than state-of-the-art neural networks. For example, on MNIST, PMM reaches

97.4% accuracy with fewer than 5,000 parameters, outperforming or matching several deep neural networks that require hundreds of thousands of weights. Introducing learnable convolutional layers before the PMM (ConvPMM) further improves performance, achieving 99.1% on MNIST and 90.9% on Fashion-MNIST, while maintaining a dramatically lower parameter count than conventional CNNs. Notably, most parameters in ConvPMM originate from the convolutional layers rather than the PMM core itself.

Dataset	Model	Accuracy (%)	Float Ops
MNIST	DNN-2	96.5	~311,650
	DNN-3	97.0	~386,718
	DNN-5	97.2	~575,050
	PMM	**97.38**	**4,990**
	GECCO	98.04	~19,000
	CTM-250	98.82	31,750
	ConvPMM	**98.99**	**129,416**
	CTM-8000	99.4	527,250
	Eff.-CapsNet	99.84	161,824
Fashion MNIST	GECCO	88.09	~19,000
	CTM-250	88.09	31,750
	PMM	**88.58**	**16,744**
	ConvPMM	**90.89**	**278,280**
	CTM-8000	91.5	~7,300,000
	MLP	91.63	~3,200,000
	VGG8b	95.47	129,416
	F-T DARTS	96.91	527,250
EMNIST Balanced	CNN	79.61	21,840
	PMM	**81.57**	**13,792**
	CNN (S-FC)	82.77	13,820
	CNN (S-FC)	83.21	16,050
	HM2-BP	85.57	665,647
	ConvPMM	**85.95**	**349,172**

Table 6.1 We show PMM and ConvPMM results compared to other highly efficient methods on several image classification datasets. We present the accuracy on withheld test sets in percent and the number of trainable floating point parameters. Results obtained from from [23]

To further evaluate scalability, Ref. [23] also explored PMMs on more challenging datasets such as CIFAR-10 and Caltech-101. In those studies, the PMM replaced the classification head of a pre-trained ResNet-50, demonstrating seamless hybrid compatibility between PMMs and existing

deep learning architectures.

Comprehensive experimental details, including PMM implementation, training protocols, hyperparameters, and additional datasets, are provided in Ref. [23].

6.7.2.3 Summary

Across both regression and classification tasks, the Parametric Matrix Model (PMM) demonstrated strong capability as a general-purpose function approximator, achieving high accuracy while maintaining remarkable parameter efficiency. In all benchmarks, PMMs required one to two orders of magnitude fewer trainable parameters than comparable state-of-the-art neural networks, yet matched or exceeded their performance. Although these represent the first applications of the PMM framework, the results already highlight its potential for future architectural extensions and domain-specific adaptations.

A particularly important implication is the computational efficiency of PMMs: their training and inference cost scales comparably to that of different neural network architectures [23], but with dramatically reduced parameter counts. This efficiency translates directly to lower energy consumption and faster execution, key advantages in an era where large-scale models, such as modern chatbots, demand enormous computational resources. Beyond environmental and operational benefits, the lightweight and mathematically structured nature of PMMs, rooted in linear algebra, also enhances their interpretability. This combination of speed, efficiency, and transparency positions PMMs as a promising alternative for real-time prediction, embedded applications, and sustainable large-scale AI.

CHAPTER 7

OUTLOOK

This thesis introduced and developed Parametric Matrix Models (PMMs), a new class of machine learning algorithms inspired by the mathematical structures of physical systems. By framing learning problems through matrix equations and eigen-decompositions, PMMs depart from conventional neural network paradigms to offer an interpretable, efficient, and physically grounded alternative. Across applications ranging from nuclear physics to regression and classification benchmarks, PMMs consistently demonstrated strong accuracy, generalization, and parameter efficiency, often matching or exceeding state-of-the-art neural networks while using one to two orders of magnitude fewer trainable parameters. These results highlight PMMs as promising building blocks for a new generation of scientific machine learning tools that combine the rigor of physical modeling with the flexibility of data-driven learning.

The success of PMMs in this work underscores their broader potential as reduced-order models capable of learning effective representations of complex systems directly from data. In nuclear many-body physics, PMMs learned effective Hamiltonians from limited observables, capturing nonperturbative effects inaccessible to conventional perturbative methods. In quantum computing, they mitigated Trotter errors by extrapolating to the continuous-time limit, improving accuracy on noisy intermediate-scale (NISQ) devices. And in fluid dynamics, PMMs embedded physical constraints directly within reduced-order formulations, enabling efficient and stable surrogate modeling. Taken together, these results demonstrate the versatility of PMMs as interpretable and computationally efficient approximators that retain fidelity to the underlying physics.

Looking forward, there are several natural directions for extending this work. A key next step is applying PMMs to larger-scale *ab initio* nuclear systems, particularly those governed by chiral effective field theory interactions. Extending the framework to heavier nuclei will test its scalability and provide a pathway toward realistic predictive modeling in regimes inaccessible to traditional methods. Similarly, integrating PMMs with hybrid quantum–classical algorithms offers a promising frontier: quantum devices can generate observables, while PMMs learn effective

operators and extrapolate beyond hardware limitations, potentially accelerating progress toward practical quantum simulations.

PMMs can be expanded into real-time and nonequilibrium modeling, enabling applications in reaction dynamics, transport, and control. Extending PMMs to handle time-dependent data and oscillatory signals will open new possibilities for online learning and real-time emulation. Additionally, hybrid architectures that combine PMMs with deep learning components, such as convolutional or attention-based modules, could capture high-dimensional features while preserving interpretability and efficiency. These architectures may bridge the gap between structured physical models and flexible neural networks, offering scalable yet explainable learning frameworks.

Another important area is uncertainty quantification and Bayesian inference. Developing systematic approaches to estimate and propagate uncertainties through PMMs would enhance their role as fast, trustworthy emulators for parameter estimation and experimental design. Coupled with their low computational cost, this makes PMMs attractive for scenarios requiring repeated evaluations, such as Monte Carlo sampling or probabilistic forecasting.

In summary, this thesis lays the foundation for a broad research program centered on Parametric Matrix Models as interpretable, efficient, and physically principled alternatives to modern deep learning. Their combination of mathematical structure, computational efficiency, and generality positions PMMs to make a lasting impact across scientific computing, quantum technologies, and machine learning. The work presented here represents a beginning rather than an endpoint, one that invites continued exploration into new architectures, domains, and theoretical connections that will shape the next generation of physics-informed AI.

APPENDIX

POD-PMM NAVIER-STOKES

A.1 PMM for Navier-Stokes

First and formost we must contruct the proper orthogonal decomposition (POD) solution for the Navier-Stokes equation, before defining the PMM.

Given snapshots of $\psi \in^{n\times N\times d+1}$ (from high-fidelity code/experiment), where $\psi = (u_x, u_y, u_z, p)$ and u_x, u_y, u_z representing the velocity components in $3D$ space, and p prepresenting pressure. We can project the data down to $\phi \in^{m\times N\times d+1}$, where $\phi = (v_x, v_y, v_z, q)$. The projection is accomplished by keeping all components of ψ in their respective spaces through the application of the reduced SVD on each component (u_x, u_y, u_z, p) individually to obtain the projection matrices $P_x, P_y, P_z, P_p \in^{m\times n}$ by selecting the first $\overline{m}$ columns of the left singular vectors U_x, U_y, U_z, U_p. We then project ψ down to ϕ using the respective projection matrices for each component:

$$\begin{aligned} v_x &= P_x^T u_x \\ v_y &= P_y^T u_y \\ v_z &= P_z^T u_z \\ q &= P_p^T p \end{aligned} \tag{.1}$$

A.2 2-D Navier-Stokes: POD

A.2.1 Force

The POD for the force term is given by the following steps:

$$u(t) = u_0 + \Delta t F \tag{.2}$$

$$\begin{aligned} u_x(t) &= P_x v_{x0} + \Delta t P_x f_x & u_y(t) &= P_y v_{y0} + \Delta t P_y f_y \\ P_x^T u_x(t) &= P_x^T P_x v_{x0} + \Delta t P_x^T P_x f_x & P_y^T u_y(t) &= P_y^T P_y v_{y0} + \Delta t P_y^T P_y f_y \\ v_x(t) &= v_{x0} + \Delta t f_x & v_y(t) &= v_{y0} + \Delta t f_y \end{aligned} \tag{.3}$$

A.2.2 Advection Term

The advection term in the Navier-Stokes equation is given by the Hadamard product between the velocity field $\mathbf{u}$ and the gradient operator ∇. Here we consider the skew-symmetric form of the advecation term:

$$(\mathbf{u}\cdot\nabla)\mathbf{u} = \frac{1}{2}\mathbf{u}\cdot\nabla\mathbf{u} + \frac{1}{2}\nabla\cdot(\mathbf{uu})$$

$$(\mathbf{u}\cdot\nabla)\mathbf{u} = \begin{bmatrix} u^x \odot \nabla^x u^x + u^y \odot \nabla^y u^x \\ u^x \odot \nabla^x u^y + u^y \odot \nabla^y u^y \end{bmatrix} \tag{.4}$$

$$\nabla\cdot(\mathbf{uu}) = \begin{bmatrix} \nabla_x u_x \odot u_x + \nabla_y u_x \odot u_y \\ \nabla_x u_y \odot u_x + \nabla_y u_y \odot u_y \end{bmatrix} \tag{.5}$$

Here, the Hadamard product is defined element-wise in the original space. Let's examine each term in the advection expression individually in the lower dimensional POD space. First lets look at $\mathbf{u}\odot\nabla\mathbf{u}$:

$$\begin{aligned}
u^x \odot \nabla^x u^x &= u_i^x \nabla_{ij}^x u_j^x & u^y \odot \nabla^y u^y &= u_i^y \nabla_{ij}^y u_j^y \\
&= P_{ir}^x v_r^x P_{il}^x g_{lt}^x P_{tj}^{xT} P_{jd}^x v_d^x & &= P_{ir}^y v_r^y P_{il}^y g_{lt}^y P_{tj}^{yT} P_{jd}^y v_d^y \\
&= P_{ir}^x v_r^x P_{il}^x g_{lt}^x \delta_{td} v_d^x & &= P_{ir}^y v_r^y P_{il}^y g_{lt}^y \delta_{td} v_d^y \\
&= P_{ir}^x v_r^x P_{il}^x g_{ld}^x v_d^x & &= P_{ir}^y v_r^y P_{il}^y g_{ld}^y v_d^y \\
&= P_{ir}^x P_{il}^x v_r^x g_{ld}^x v_d^x & &= P_{ir}^y P_{il}^y v_r^y g_{ld}^y v_d^y \\
v^x \odot g^x v^x &= P_{qi}^{xT} P_{ir}^x P_{il}^x v_r^x g_{ld}^x v_d^x & v^y \odot g^y v^y &= P_{qi}^{yT} P_{ir}^y P_{il}^y v_r^y g_{ld}^y v_d^y \\
&= P_{qi}^{xT} P_{ir}^x P_{il}^x v_r^x g_{ld}^x v_d^x & &= P_{qi}^{yT} P_{ir}^y P_{il}^y v_r^y g_{ld}^y v_d^y \\
&= \mathbb{T}_{q|rl}^x v_r^x g_{ld}^x v_d^x & &= \mathbb{T}_{q|rl}^y v_r^y g_{ld}^y v_d^y
\end{aligned} \tag{.6}$$

$$
\begin{aligned}
u^y \odot \nabla^y u^x &= u_i^y \nabla_{ij}^y u_j^x & u^x \odot \nabla^x u^y &= u_i^x \nabla_{ij}^x u_j^y \\
&= P_{ir}^x v_r^y P_{il}^x g_{lt}^y P_{tj}^{xT} P_{jd}^x v_d^x & &= P_{ir}^y v_r^x P_{il}^y g_{lt}^x P_{tj}^{yT} P_{jd}^y v_d^y \\
v^y \odot g^y v^x &= P_{qi}^{xT} P_{ir}^x v_r^y P_{il}^x g_{lt}^y \delta_{td} v_d^x & v^x \odot g^x v^y &= P_{qi}^{yT} P_{ir}^y P_{il}^y v_r^x g_{lt}^x \delta_{td} v_d^y \\
&= P_{qi}^{xT} P_{ir}^x P_{il}^x v_r^y g_{ld}^y v_d^x & &= P_{qi}^{yT} P_{ir}^y P_{il}^y v_r^x g_{ld}^x v_d^y \\
&= \mathbb{T}_{q|rl}^x v_r^y g_{ld}^y v_d^x & &= \mathbb{T}_{q|rl}^y v_r^x g_{ld}^x v_d^y
\end{aligned}
\tag{.7}
$$

Note that the Hadamard product is only defined in the higher-dimensional space due to the presence of the projection operators. This means one must first map the data to the higher-dimensional space, perform the Hadamard product, and then project it back down. However, directly interacting with the higher-dimensional space during the learning process is inefficient. Ideally, we want to stay within the latent space at all times. To address this, we can pre-contract the projection operators that define the Hadamard product into a rank-3 tensor $\mathbb{T}_{q|rl} \in^{\overline{m}\times\overline{m}\times\overline{m}}$, thereby avoiding the need for higher-dimensional operations during learning. Next we look at the divergence term $\nabla \mathbf{u} \odot \mathbf{u}$:

$$
\begin{aligned}
\nabla_x u_x \odot u_x &= \nabla_{ij}^x u_j^x u_j^x & \nabla_y u_y \odot u_y &= \nabla_{ij}^y u_j^y u_j^y \\
&= P_{ik}^x g_{kq}^x P_{qj}^{xT} P_{jr}^x v_r^x P_{jl}^x v_l^x & &= P_{ik}^y g_{kq}^y P_{qj}^{yT} P_{jr}^y v_r^y P_{jl}^y v_l^y \\
&= P_{ik}^x P_{qj}^{xT} P_{jr}^x P_{jl}^x g_{kq}^x v_r^x v_l^x & &= P_{ik}^y P_{qj}^{yT} P_{jr}^y P_{jl}^y g_{kq}^y v_r^y v_l^y \\
g_x v_x \odot v_x &= P_{pi}^{xT} P_{ik}^x P_{qj}^{xT} P_{jr}^x P_{jl}^x g_{kq}^x v_r^x v_l^x & g_y v_y \odot v_y &= P_{pi}^{yT} P_{ik}^y P_{qj}^{yT} P_{jr}^y P_{jl}^y g_{kq}^y v_r^y v_l^y \\
&= g_{kq}^x \mathbb{T}_{q|rl}^x v_r^x v_l^x & &= g_{kq}^y \mathbb{T}_{q|rl}^y v_r^y v_l^y
\end{aligned}
\tag{.8}
$$

$$
\begin{aligned}
\nabla_y u_x \odot u_y &= \nabla_{ij}^y u_j^x u_j^y & \nabla_x u_y \odot u_x &= \nabla_{ij}^x u_j^y u_j^x \\
&= P_{ik}^x g_{kq}^y P_{qj}^{xT} P_{jr}^x v_r^x P_{jl}^x v_l^y & &= P_{ik}^y g_{kq}^x P_{qj}^{yT} P_{jr}^y v_r^y P_{jl}^y v_l^x \\
g_y v_x \odot v_y &= P_{pi}^{xT} P_{ik}^x g_{kq}^y P_{qj}^{xT} P_{jr}^x v_r^x P_{jl}^x v_l^y & g_x v_y v_x &= P_{pi}^{yT} P_{ik}^y g_{kq}^x P_{qj}^{yT} P_{jr}^y v_r^y P_{jl}^y v_l^x \\
&= \delta_{pk} g_{kq}^y P_{qj}^{xT} P_{jr}^x P_{jl}^x v_r^x v_l^y & g_x v_y v_x &= \delta_{pk} g_{kq}^x P_{qj}^{yT} P_{jr}^y P_{jl}^y v_r^y v_l^x \\
&= g_{pq}^y \mathbb{T}_{q|rl}^x v_r^x v_l^y & &= g_{pq}^x \mathbb{T}_{q|rl}^y v_r^y v_l^x
\end{aligned}
\tag{.9}
$$

A.2.3 Diffusion

First lets examine the diffusion term in the Navier-Stokes equation:

$$\begin{aligned} \frac{\partial u}{\partial t} &= \nu \nabla^2 u \\ u(t) &= u_0 + \nu \Delta t \nabla^2 u_0 \\ u(t) &= u_0 + \nu \Delta t \nabla_x^2 u_0 + \nu \Delta t \nabla_y^2 u_0 \end{aligned} \tag{.10}$$

Now we look at how the POD solution looks like

$$u_x(t) = u_{x0} + \nu \Delta t \nabla_x^2 u_{x0} + \nu \Delta t \nabla_y^2 u_{x0} \qquad u_y(t) = u_{y0} + \nu \Delta t \nabla_x^2 u_{y0} + \nu \Delta t \nabla_y^2 u_{y0}$$

$$u_x(t) = P_x v_{x0} + \nu \Delta t P_x g_x^2 P_x^T P_x v_{x0} + \nu \Delta t P_x g_y^2 P_x^T P_x v_{x0} \qquad u_y(t) = P_y v_{y0} + \nu \Delta t P_y g_x^2 P_y^T P_y v_{y0} + \nu \Delta t P_y g_y^2 P_y^T P_y v_{y0}$$

$$u_x(t) = P_x v_{x0} + \nu \Delta t P_x g_x^2 v_{x0} + \nu \Delta t P_x g_y^2 v_{x0} \qquad u_y(t) = P_y v_{y0} + \nu \Delta t P_y g_x^2 v_{y0} + \nu \Delta t P_y g_y^2 v_{y0}$$

$$v_x(t) = P_x^T P_x v_{x0} + \nu \Delta t P_x^T P_x g_x^2 v_{x0} + \nu \Delta t P_x^T P_x g_y^2 v_{x0} \qquad v_y(t) = P_y^T P_y v_{y0} + \nu \Delta t P_y^T P_y g_x^2 v_{y0} + \nu \Delta t P_y^T P_y g_y^2 v_{y0}$$

$$v_x(t) = v_{x0} + \nu \Delta t g_x^2 v_{x0} + \nu \Delta t g_y^2 v_{x0} \qquad v_y(t) = v_{y0} + \nu \Delta t g_x^2 v_{y0} + \nu \Delta t g_y^2 v_{y0}$$

$$v_x(t) = v_{x0} + \nu \Delta t g^2 v_{x0} \qquad v_y(t) = v_{y0} + \nu \Delta t g^2 v_{y0}$$

A.2.4 Pressure Correction

The pressure correction term in the Navier-Stokes equation is given by the following steps:

$$\begin{aligned} \frac{\partial u}{\partial t} &= -\frac{1}{\rho} \nabla p \\ u(t) &= u_0 - \frac{\Delta t}{\rho} \nabla p \end{aligned} \tag{.11}$$

Where the pressure field p is computed by solving the Poisson equation:

$$\begin{aligned} 0 = \nabla \cdot u(t) &= \nabla \cdot (u_0 - \frac{\Delta t}{\rho} \nabla p) \\ 0 &= \nabla \cdot u_0 - \frac{\Delta t}{\rho} \nabla \cdot \nabla p \\ \nabla^2 p &= \frac{\rho}{\Delta t} \nabla \cdot u_0 \end{aligned} \tag{.12}$$

The POD solution for the pressure correction term is given by the following steps:

$$
\begin{aligned}
\nabla^2 p &= \frac{\rho}{\Delta t}\nabla \cdot u_0 \\
(\nabla_x^2 + \nabla_y^2)p &= \frac{\rho}{\Delta t}(\nabla_x u_{x0} + \nabla_y u_{y0}) \\
p &= \frac{\rho}{\Delta t}(\nabla_x^2 + \nabla_y^2)^{-1}(\nabla_x u_{x0} + \nabla_y u_{y0}) \\
p &= \frac{\rho}{\Delta t}(P_p g_x^2 P_p^T + P_p g_y^2 P_p^T)^{-1}(P_p g_x P_p^T P_p v_{x0} + P_p g_y P_p^T P_p v_{y0}) \\
p &= \frac{\rho}{\Delta t}(P_p g_x^2 P_p^T + P_p g_y^2 P_p^T)^{-1}(P_p g_x v_{x0} + P_p g_y v_{y0}) \\
p &= \frac{\rho}{\Delta t}P_p(g_x^2 + g_p^2)^{-1}P_p^T P_p(g_x v_{x0} + g_y v_{y0}) \\
q &= \frac{\rho}{\Delta t}P_p^T P_p(g^2)^{-1}(g_x v_{x0} + g_y v_{y0}) \\
q &= \frac{\rho}{\Delta}(g^2)^{-1}(g_x v_{x0} + g_y v_{y0})
\end{aligned}
\tag{.13}
$$

Then we can compute the gradient of the pressure field q in the lower dimensional space:

$$
\begin{aligned}
\nabla p &= \begin{bmatrix} \nabla_x p \\ \nabla_y p \end{bmatrix} \\
&= \begin{bmatrix} P_x g_x P_x^T P_x q \\ P_y g_y P_y^T P_x q \end{bmatrix} \\
gq &= \begin{bmatrix} P_x^T P_x g_x q \\ P_y^T P_y g_y q \end{bmatrix} \\
gq &= \begin{bmatrix} g_x q \\ g_y q \end{bmatrix}
\end{aligned}
\tag{.14}
$$

Resulting in the pressure correction term in the lower dimensional space:

$$
\begin{aligned}
u(t) &= u_0 - \frac{\Delta t}{\rho}\nabla p \\
v(t) &= v_0 - \frac{\Delta t}{\rho} gq
\end{aligned}
\tag{.15}
$$

A.3 PMM algorithm

With the POD solution for the Navier-Stokes equation, we can now define the PMM algorithm

1) Add force term

$$w^1(t) = \begin{bmatrix} v_x(t) + \Delta t f_x \\ v_y(t) + \Delta t f_y \end{bmatrix} \tag{.16}$$

2) Apply advection term

$$\begin{aligned} w^2(t) = w^1(t) &- \frac{\Delta t}{2} \begin{bmatrix} \mathbb{T}^x_{q|rl} w_r^{1,x} \underline{g}^x_{ld} w_d^{1,x} + \mathbb{T}^x_{q|rl} w_r^{1,y} \underline{g}^y_{ld} w_d^{1,x} \\ \mathbb{T}^y_{q|rl} w_r^{1,x} \underline{g}^x_{ld} w_d^{1,y} + \mathbb{T}^y_{q|rl} w_r^{1,y} \underline{g}^y_{ld} w_d^{1,y} \end{bmatrix} \\ &- \frac{\Delta t}{2} \begin{bmatrix} \underline{g}^x_{kq} \mathbb{T}^x_{q|rl} w_r^{1,x} w_l^{1,x} + \underline{g}^y_{kq} \mathbb{T}^x_{q|rl} w_r^{1,x} w_l^{1,y} \\ \underline{g}^x_{kq} \mathbb{T}^y_{q|rl} w_r^{1,y} w_l^{1,x} + \underline{g}^y_{kq} \mathbb{T}^y_{q|rl} w_r^{1,y} w_l^{1,y} \end{bmatrix} \end{aligned} \tag{.17}$$

3) Diffuse

$$\begin{aligned} w^3(t) &= w^2(t) + \nu \Delta t \begin{bmatrix} \underline{g}^2_x w_x^2 + \underline{g}^2_y w_x^2 \\ \underline{g}^2_x w_y^2 + \underline{g}^2_y w_y^2 \end{bmatrix} \\ w^3(t) &= w^2(t) + \nu \Delta t \underline{g}^2 w^2(t) \end{aligned} \tag{.18}$$

4) Lastly apply the pressure correction term

$$\begin{aligned} w^4(t) &= w^3(t) - \frac{\Delta t}{\rho} \begin{pmatrix} g_x [(\underline{g}^2_x + \underline{g}^2_y)^{-1} (\underline{g}_x w_x^3 + \underline{g}_y w_y^3)] \\ g_y [(\underline{g}^2_x + \underline{g}^2_y)^{-1} (\underline{g}_x w_x^3 + \underline{g}_y w_y^3)] \end{pmatrix} \\ w^4(t) &= w^3(t) - \frac{\Delta t}{\rho} \underline{g} [(\underline{g}^2)^{-1} \underline{g} \cdot w^3(t)] \end{aligned} \tag{.19}$$

5) set $v(t + \Delta t) = w^4(t)$

6) learn $\mathcal{L} = \frac{1}{N} ||u(t) - Pv(t)||^2$, where $u(t)$ is the high-fedility data, and $v(t)$ is the PMM output.

The learnable parameters in the PMM algorithm are the operators that are not given. So the learnable parameters are $\underline{g}_x$ and $\underline{g}_y$, which are $\overline{m} \times \overline{m}$ matrices. If the original space is to large and computing the rank-3 tensors is too expensive or snapshots are not given, we can learn the $\overline{m} \times \overline{m} \times \overline{m}$ tensor $\underline{\mathbb{T}}$.

BIBLIOGRAPHY

[1] Daniel S. Abrams and Seth Lloyd. Simulation of many body Fermi systems on a universal quantum computer. *Phys. Rev. Lett.*, 79:2586–2589, 1997.

[2] Cassandra L. Armstrong, Pablo Giuliani, Kyle Godbey, Rahul Somasundaram, and Ingo Tews. Emulators for scarce and noisy data II: Application to auxiliary-field diffusion Monte Carlo for neutron matter. 2 2025.

[3] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[4] Mari Carmen Bañuls. Tensor network states: Basic introduction. `https://drive.google.com/file/d/1FOWJwbwyErND7XCjF5fyQ1bGHGrKTaif/view`, 2023. Lecture notes, dated March 3, 2023. Accessed: 2025-10-07.

[5] Max Bee-Lindgren, Zhengrong Qian, Matthew DeCross, Natalie C. Brown, Christopher N. Gilbreth, Jacob Watkins, Xilin Zhang, and Dean Lee. Rodeo Algorithm with Controlled Reversal Gates, 8 2022.

[6] Vaishali Bhatia and KR Ramkumar. An efficient quantum computing technique for cracking rsa using shor's algorithm. In *2020 IEEE 5th international conference on computing communication and automation (ICCCA)*, pages 89–94. IEEE, 2020.

[7] Bugra Borasoy, Evgeny Epelbaum, Hermann Krebs, Dean Lee, and U G Meißner. Two-particle scattering on the lattice: Phase shifts, spin-orbit coupling, and mixing angles. *The European Physical Journal A*, 34:185–196, 2007.

[8] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[9] Jacob C Bridgeman and Christopher T Chubb. Hand-waving and interpretive dance: an introductory course on tensor networks. *Journal of physics A: Mathematical and theoretical*, 50(22):223001, 2017.

[10] Steven L Brunton and J Nathan Kutz. Promising directions of machine learning for partial differential equations. *Nature Computational Science*, 4(7):483–494, 2024.

[11] Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738, 2021.

[12] Petr Cársky, Josef Paldus, and Jirí Pittner. Recent progress in coupled cluster methods: Theory and applications. 2010.

[13] Garnet Kin-Lic Chan, Jonathan J Dorando, Debashree Ghosh, Johannes Hachmann, Eric Neuscamman, Haitao Wang, and Takeshi Yanai. An introduction to the density matrix renormalization group ansatz in quantum chemistry. In *Frontiers in quantum systems in chemistry and physics*, pages 49–65. Springer, 2008.

[14] Garnet Kin-Lic Chan and Sandeep Sharma. The density matrix renormalization group in quantum chemistry. *Annual review of physical chemistry*, 62(1):465–481, 2011.

[15] Andrew M Childs, Yuan Su, Minh C Tran, Nathan Wiebe, and Shuchen Zhu. Theory of Trotter error with commutator scaling. *Physical Review X*, 11(1):011020, 2021.

[16] Jaeho Choi and Joongheon Kim. A tutorial on quantum approximate optimization algorithm (qaoa): Fundamentals and applications. In *2019 international conference on information and communication technology convergence (ICTC)*, pages 138–142. IEEE, 2019.

[17] Kenneth Choi, Dean Lee, Joey Bonitati, Zhengrong Qian, and Jacob Watkins. Rodeo algorithm for quantum computing. *Physical Review Letters*, 127(4):040505, 2021.

[18] Kenneth Choi, Dean Lee, Joey Bonitati, Zhengrong Qian, and Jacob Watkins. Rodeo Algorithm for Quantum Computing. *Phys. Rev. Lett.*, 127(4):040505, 2021.

[19] Pi-Yueh Chuang and Lorena A Barba. Predictive limitations of physics-informed neural networks in vortex shedding. *arXiv preprint arXiv:2306.00230*, 2023.

[20] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: an extension of MNIST to handwritten letters. Preprint at `https://arxiv.org/abs/1702.05373`, 2017.

[21] Thomas D. Cohen and Hyunwoo Oh. Optimizing the rodeo projection algorithm. *Phys. Rev. A*, 108(3):032422, 2023.

[22] Patrick Cook. Personal communication, 2025.

[23] Patrick Cook, Danny Jammooa, Morten Hjorth-Jensen, Daniel D Lee, and Dean Lee. Parametric matrix models. *Nature Communications*, 16(1):5929, 2025.

[24] Patrick Cook, Danny Jammooa, Dean Lee, Morten Hjorth-Jensen, and Daniel Lee. Parametric matrix models: Equations as data. In *APS Division of Nuclear Physics Meeting Abstracts*, volume 2025, 2025.

[25] Ryan Curry, Kai Hebeler, Stefano Gandolfi, Alexandros Gezerlis, Achim Schwenk, Rahul Somasundaram, and Ingo Tews. Quantum monte carlo calculations of light nuclei with fully propagated theoretical uncertainties, 2025.

[26] Ryan Curry, Jasmine Kozar, and Alexandros Gezerlis. Ab initio study of the neutron and fermi polarons on the lattice. *arXiv preprint arXiv:2510.05233*, 2025.

[27] Alexander Del Toro Barba. Simulating quantum systems with qubitization, trotterization and linear combination of unitaries (lcu), September 2024. Medium article.

[28] Uwe Dorner, Rafal Demkowicz-Dobrzanski, Brian J Smith, Jeff S Lundeen, Wojciech Wasilewski, Konrad Banaszek, and Ian A Walmsley. Optimal quantum phase estimation. *Phys. Rev. Lett.*, 102(4):040403, 2009.

[29] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A—Atomic, Molecular, and Optical Physics*, 86(3):032324, 2012.

[30] Dillon Frame, Rongzheng He, Ilse Ipsen, Daniel Lee, Dean Lee, and Ermal Rrapaj. Eigenvector continuation with subspace learning. *Physical review letters*, 121(3):032501, 2018.

[31] J. Gamito, J. Khalouf-Rivera, J. M. Arias, P. Pérez-Fernández, and F. Pérez-Bernal. Excited-state quantum phase transitions in the anharmonic lipkin-meshkov-glick model: Static aspects. *Phys. Rev. E*, 106:044125, Oct 2022.

[32] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.", 2022.

[33] Tudor Giurgica-Tiron, Yousef Hindy, Ryan LaRose, Andrea Mari, and William J Zeng. Digital zero noise extrapolation for quantum error mitigation. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 306–316. IEEE, 2020.

[34] Tamara G Grossmann, Urszula Julia Komorowska, Jonas Latz, and Carola-Bibiane Schönlieb. Can physics-informed neural networks beat the finite element method? *IMA Journal of Applied Mathematics*, page hxae011, 2024.

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

[36] Jutho Haegeman, Christian Lubich, Ivan Oseledets, Bart Vandereycken, and Frank Verstraete. Unifying time evolution and optimization with matrix product states. *Physical Review B*, 94(16):165116, 2016.

[37] Saad Hikmat Haji and Adnan Mohsin Abdulazeez. Comparison of optimization techniques based on gradient descent algorithm: A review. *PalArch's Journal of Archaeology of Egypt/Egyptology*, 18(4):2715–2743, 2021.

[38] W D Heiss, F G Scholtz, and H B Geyer. The large n behaviour of the lipkin model and exceptional points. *Journal of Physics A: Mathematical and General*, 38(9):1843, feb 2005.

[39] Heiko Hergert. A guided tour of ab initio nuclear many-body theory. *Frontiers in Physics*, 8:379, 2020.

[40] Heiko Hergert, JM Yao, Titus D Morris, Nathan M Parzuchowski, Scott K Bogner, and Jonathan Engel. Nuclear structure from the in-medium similarity renormalization group. In *Journal of Physics: Conference Series*, volume 1041, page 012007. IOP Publishing, 2018.

[41] L Jin, A Ravlić, P Giuliani, K Godbey, and W Nazarewicz. Surrogate models for linear response. *arXiv preprint arXiv:2510.13989*, 2025.

[42] Bhavana Jonnalagadda and Stephen Becker. Evaluation of data driven low-rank matrix factorization for accelerated solutions of the vlasov equation. *arXiv preprint arXiv:2501.04024*, 2024.

[43] Tosio Kato. *Perturbation theory for linear operators*, volume 132. Springer Science & Business Media, 2013.

[44] Jane Mee Kim. *Solving the Quantum Many-Body Problem with Neural-Network Quantum States*. Michigan State University, 2023.

[45] Youngseok Kim, Andrew Eddins, Sajant Anand, Ken Xuan Wei, Ewout Van Den Berg, Sami Rosenblatt, Hasan Nayfeh, Yantao Wu, Michael Zaletel, Kristan Temme, et al. Evidence for the utility of quantum computing before fault tolerance. *Nature*, 618(7965):500–505, 2023.

[46] A. Yu. Kitaev. Quantum measurements and the Abelian stabilizer problem. *Electronic Colloquium on Computational Complexity*, 3, 1995.

[47] Emanuel Knill and Raymond Laflamme. Concatenated quantum codes. *arXiv preprint quant-ph/9608012*, 1996.

[48] Felix Koehler. Machine Learning and Simulation. `https://github.com/ceyron/machine-learning-and-simulation`. DOI: 10.5281/zenodo.12793323.

[49] Hannah Lange, Anka Van de Walle, Atiye Abedinnia, and Annabelle Bohrdt. From architectures to applications: A review of neural quantum states. *Quantum Science and Technology*, 2024.

[50] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[51] Dean Lee. Lattice effective field theory simulations of nuclei. *Annual Review of Nuclear and Particle Science*, 75, 2025.

[52] Jing Liu, Sujie Li, Jiang Zhang, and Pan Zhang. Tensor networks for unsupervised machine learning. *Physical Review E*, 107(1):L012103, 2023.

[53] Alessandro Lovato, Corey Adams, Giuseppe Carleo, and Noemi Rocco. Hidden-nucleons neural-network quantum states for the nuclear many-body problem. *Physical Review Research*, 4(4):043178, 2022.

[54] Bing-Nan Lu, Timo A. Lähde, Dean Lee, and Ulf-G. Meißner. Precise determination of lattice phase shifts and mixing angles. *Physics Letters B*, 760:309–313, 2016.

[55] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

[56] Timo A Lähde, Evgeny Epelbaum, Hermann Krebs, Dean Lee, Ulf-G Meißner, and Gautam Rupak. Uncertainties of euclidean time extrapolation in lattice effective field theory. *Journal of Physics G: Nuclear and Particle Physics*, 42(3):034012, feb 2015.

[57] Andrea Mari, Nathan Shammah, and William J Zeng. Extending quantum probabilistic error cancellation by noise scaling. *Physical Review A*, 104(5):052607, 2021.

[58] Nick McGreivy and Ammar Hakim. Weak baselines and reporting biases lead to overoptimism in machine learning for fluid-related partial differential equations. *Nature Machine Intelligence*, pages 1–14, 2024.

[59] Hesameddin Mohammadi, Meisam Razaviyayn, and Mihailo R Jovanović. Robustness of accelerated first-order algorithms for strongly convex optimization problems. *IEEE Transactions on Automatic Control*, 66(6):2480–2495, 2020.

[60] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

[61] MS Moreira, Gian Giacomo Guerreschi, Wouter Vlothuizen, Jorge F Marques, Jeroen van Straten, Shavindra P Premaratne, Xiang Zou, Hany Ali, Nandini Muthusubramanian, Christos Zachariadis, et al. Realization of a quantum neural network using repeat-until-success circuits in a superconducting quantum processor. *npj Quantum Information*, 9(1):118, 2023.

[62] Mario Motta, Chong Sun, Adrian TK Tan, Matthew J O'Rourke, Erika Ye, Austin J Minnich, Fernando GSL Brandao, and Garnet Kin-Lic Chan. Determining eigenstates and thermal states on a quantum computer using quantum imaginary time evolution. *Nature Physics*, 16(2):205–210, 2020.

[63] Daniel Naegels. An introduction to goldstone boson physics and to the coset construction. *arXiv preprint arXiv:2110.14504*, 2021.

[64] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[65] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 2006.

[66] Daniel Odell, Pablo Giuliani, Kyle Beyer, Manuel Catacora-Rios, MY-H Chan, Edgard Bonilla, Richard J Furnstahl, Kyle Godbey, and Filomena M Nunes. Rose: A reduced-order scattering emulator for optical models. *Physical Review C*, 109(4):044612, 2024.

[67] US Department of Energy (USDOE). A new era of discovery: The 2023 long range plan for nuclear science. Technical report, US Department of Energy (USDOE), Washington, DC (United States). Office of Science, 10 2023.

[68] Feng Pan, Keyang Chen, and Pan Zhang. Solving the sampling problem of the sycamore quantum circuits. *Physical Review Letters*, 129(9):090502, 2022.

[69] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5(1):4213, 2014.

[70] Alexander N Prokopenya. Simulation of a quantum algorithm for phase estimation. *Programming and Computer Software*, 41(2):98–104, 2015.

[71] Zhengrong Qian, Jacob Watkins, Gabriel Given, Joey Bonitati, Kenneth Choi, and Dean Lee. Demonstration of the Rodeo Algorithm on a Quantum Computer, 10 2021.

[72] Shi-Ju Ran, Emanuele Tirrito, Cheng Peng, Xi Chen, Gang Su, and Maciej Lewenstein. Review of tensor network contraction approaches. *arXiv preprint arXiv:1708.09213*, 2017.

[73] Susmita Ray. A quick review of machine learning algorithms. In *2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon)*, pages 35–39. IEEE, 2019.

[74] Gumaro Rendon, Jacob Watkins, and Nathan Wiebe. Improved accuracy for trotter simulations using chebyshev interpolation. *Quantum*, 8:1266, 2024.

[75] Peter Ring and Peter Schuck. *The nuclear many-body problem*. Springer Science & Business Media, 2004.

[76] Raul Rojas. The backpropagation algorithm. In *Neural networks: a systematic introduction*, pages 149–182. Springer, 1996.

[77] Edgar Andres Ruiz Guzman and Denis Lacroix. Accessing ground-state and excited-state energies in a many-body system after symmetry restoration using quantum computers. *Phys. Rev. C*, 105(2):024324, 2022.

[78] AC Semposki, C Drischler, RJ Furnstahl, JA Melendez, and DR Phillips. From chiral effective field theory to perturbative qcd: A bayesian model mixing approach to symmetric nuclear matter. *Physical Review C*, 111(3):035804, 2025.

[79] Alexandra Catherine Semposki. *Bayesian Model Mixing and Uncertainty Quantification for the Dense Matter Equation of State*. PhD thesis, Ohio University, 2025.

[80] Richik Sengupta, Soumik Adhikary, Ivan Oseledets, and Jacob Biamonte. Tensor networks in machine learning. *European Mathematical Society Magazine*, (126):4–12, 2022.

[81] Peter W Shor. Introduction to quantum algorithms. In *Proceedings of Symposia in Applied Mathematics*, volume 58, pages 143–160, 2002.

[82] Rahul Somasundaram, Cassandra L. Armstrong, Pablo Giuliani, Kyle Godbey, Stefano Gandolfi, and Ingo Tews. Emulators for scarce and noisy data: application to auxiliary field diffusion Monte Carlo for the deuteron. 4 2024.

[83] Rahul Somasundaram, Isak Svensson, Soumi De, Andrew E. Deneris, Yannick Dietz, Philippe Landry, Achim Schwenk, and Ingo Tews. Inferring three-nucleon couplings from multi-messenger neutron-star observations. 9 2024.

[84] Jos Stam. Stable fluids. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 779–786. 2023.

[85] Edwin Stoudenmire and David J Schwab. Supervised learning with tensor networks. *Advances in neural information processing systems*, 29, 2016.

[86] Masuo Suzuki. Generalized Trotter's formula and systematic approximants of exponential operators and inner derivations with applications to many-body problems. *Comm. Math. Phys.*, 51(2):183–190, June 1976.

[87] Masuo Suzuki. General decomposition theory of ordered exponentials. *Proceedings of the Japan Academy, Series B*, 69(7):161–166, 1993.

[88] Krysta M. Svore, Matthew B. Hastings, and Michael Freedman. Faster Phase Estimation, 4 2013.

[89] Kristan Temme, Sergey Bravyi, and Jay M Gambetta. Error mitigation for short-depth quantum circuits. *Physical review letters*, 119(18):180509, 2017.

[90] Joseph Tindall, Matthew Fishman, E Miles Stoudenmire, and Dries Sels. Efficient tensor network simulation of ibm's eagle kicked ising experiment. *Prx quantum*, 5(1):010308, 2024.

[91] Lloyd N Trefethen and David Bau. *Numerical linear algebra*. SIAM, 2022.

[92] Hale F Trotter. On the product of semi-groups of operators. *Proceedings of the American Mathematical Society*, 10(4):545–551, 1959.

[93] Gerhard Venter. Review of optimization techniques. 2010.

[94] Steven R White. Density matrix formulation for quantum renormalization groups. *Physical review letters*, 69(19):2863, 1992.

[95] Steven R White. Density-matrix algorithms for quantum renormalization groups. *Physical review b*, 48(14):10345, 1993.

[96] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. Preprint at `https://arxiv.org/abs/1708.07747`, 2017.

[97] Bo Yang, Rudy Raymond, and Shumpei Uno. Efficient quantum readout-error mitigation for sparse measurement outcomes of near-term quantum devices. *Physical Review A*, 106(1):012423, 2022.

[98] Hang Yu and Takayuki Miyagi. An Efficient Learning Method to Connect Observables. 3 2025.

[99] Mohamed Ziouane, Hafida Hamdi, Mourad Nachaoui, and Abdeljalil Nachaoui. A deep numerical study of bfgs and lbfgs methods for solving. *Computational Methods for Inverse Problems and Applications: ICMDS 2024, Khouribga, Morocco, October 21–22*, 498:181, 2025.

[100] Alexander Zlokapa and Alexandru Gheorghiu. A deep learning model for noise prediction on near-term quantum devices. *arXiv preprint arXiv:2005.10811*, 2020.