

Autoregressive Generative Models for Many-body Physics

by

Yi Hong Teoh

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Physics

Waterloo, Ontario, Canada, 2025

© Yi Hong Teoh 2025

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: Dr. Lei Wang
 Professor,
 Institute of Physics,
 Chinese Academy of Sciences

Supervisor(s): Dr. Roger Melko
 Professor,
 Dept. of Physics and Astronomy,
 University of Waterloo
 and Associate Faculty,
 Perimeter Institute for Theoretical Physics

Dr. Kazi Rajibul Islam
Associate professor,
Institute for Quantum Computing
and Dept. of Physics and Astronomy,
University of Waterloo

Internal Member: Dr. Alan Jamison
 Assistant Professor,
 Dept. of Physics and Astronomy,
 University of Waterloo

Dr. Estelle Inack
Adjunct Faculty,
Dept. of Physics and Astronomy,
University of Waterloo

Internal-External Member: Dr. Mohammad Kohandel
Professor,
Dept. of Applied Mathematics,
University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Yi Hong Teoh was the sole author of chapters 1, 2 and 5 to 7. The content in these chapters were not written for publication. Chapters 3 and 4 consist in part of material written for publication.

For the research in chapter 2:

This work was conducted under the supervision of Dr. Roger Melko at the University of Waterloo. Yi Hong Teoh is the primary coder of the project and performed the analysis shown in this chapter. Dmitri Iouchtchenko provided discussion on the comparison between classical shadows and neural quantum states while Ejaaz Merali provided discussion and additional analysis on the conditional local density RNN.

For the research in chapter 3:

This work was conducted under the supervision of Dr. Roger Melko at the University of Waterloo. Yi Hong Teoh is the primary coder of the project and performed all analysis shown in this chapter. Ejaaz Merali and M. Schyler Moss has provided some discussion on the results demonstrated. The manuscript was written collaboratively by Yi Hong Teoh and Dr. Roger Melko.

Citation:

[1] Y. H. Teoh and R. G. Melko, Autoregressive model path dependence near ising criticality (2024), arXiv:2408.15715 [cs.LG].

For the research in chapter 4:

This work was conducted under the supervision of Dr. Roger Melko at the University of Waterloo.

Yi Hong Teoh and David Fitzek developed the model architecture and code collaboratively. Hin Pok Fung and Ejaaz Merali worked on data generation, while Benjamin MacLellan managed the storage and open publication of the data. David Fitzek and Gebremedhin A. Dagnew trained the model on the supercomputing cluster. Analysis was performed by Yi Hong Teoh, Gebremedhin A. Dagnew and M. Schuyler Moss.

The manuscript was written collaboratively by everyone above, Yi Hong Teoh mainly contributed to the writing of the description of the architecture, the presentation of the results and the conclusion.

Citation:

[2] D. Fitzek, Y. H. Teoh, H. P. Fung, G. A. Dagneu, E. Merali, M. S. Moss, B. MacLellan, and R. G. Melko, Rydberggpt (2024), arXiv:2405.21052 [quant-ph].

For the research in chapter 5:

This work was conducted under the supervision of Dr. Roger Melko at the University of Waterloo. Yi Hong Teoh is the primary coder of the project and performed all analysis shown in this chapter.

For the research in chapter 6:

This work was conducted under the supervision of Dr. Roger Melko at the University of Waterloo. Yi Hong Teoh is the primary coder of the project and performed all analysis shown in this chapter.

Abstract

Many body physics, the study of emergent behavior from the microscopic interactions between countless degrees of freedom, is fundamental to our understanding of the universe. Understanding these systems enable the design and development of everything from materials, pharmaceuticals and even machine learning algorithms. However, to our knowledge, classical simulation of these systems are naturally insufficient and or inefficient.

Recent developments in quantum processors herald a new age in the study of many body systems. The ability to process, extract and generalize the information from such devices is pertinent to new discoveries in the field.

In recent years, autoregressive generative models have been proven to have remarkable capabilities in a wide range of applications, from machine translation, text summarization to image generation. These models not only allow exact evaluation of the likelihood but also enable independent and identically distributed samples to be drawn from their encapsulated complex joint probability distribution of many degrees of freedom. Additionally, these models have demonstrated emergence, where the model exhibits complex behaviors allowing exceptional performance in a wide range of scenarios. These algorithms are prime candidates for the extraction of information from quantum processors and many-body systems.

In current times, data from quantum processors is still rare and expensive, as such we desire the most efficient method to extract information from such limited data. Generative models reveal themselves to be powerful tools in this scenario, achieving higher accuracies with little data.

Rich many-body systems typically inhabit higher dimensional spaces compared to the 1-dimensional sequence of autoregressive models. Consequently, a choice is required regarding the traversal of these higher dimensional systems. We explore the effects linked to this choice of traversal in the training of such models.

Furthermore, we systematically probe the generalization abilities of autoregressive generative models in a variety of axes, such as in size and parameters, for typical discrete and continuous many-body systems.

Finally, Inspired by the recent establishment of correspondence between machine learning and many-body physics, we integrate real-space renormalization group into the model architecture. This integration of a many-body physics technique with machine learning demonstrates considerable promise for the future development of powerful architectures capable of achieving generality.

Acknowledgements

First and foremost, I extend my sincerest gratitude to my supervisors Dr. Roger Melko and Dr. Kazi Rajibul Islam my co-supervisor. They are no doubt extremely knowledgeable supervisors with a great deal of understanding of physics. Additionally, they are amazing leaders that are easily approachable for advice. The effort they have made to make their students' life under them comfortable is greatly appreciated.

I would also like to express my appreciation to Dr. Ray Laflamme and Dr. Alan Jamison for devoting their time to attend my committee meetings and provide feedback. Their comments have improved my ability to convey my ideas.

A huge thank you to my fellow lab members in the Perimeter Institute Quantum Intelligence Lab (PIQuIL) both former and current, Benjamin MacLellan, Sehmimul Hoque, M. Schuyler Moss, Xiezhe (Roger) Luo, Gebremedhin A. Dagneu, Shiyu Zhou, Tarun Kumar, Ejaaz Merali, Dmitri Iouchtchenko, David Fitzek, Severyn Balaniuk, Yangrui Hu, Stefanie Czischek and others. Benjamin MacLellan has been a great and close collaborator on many a project, I am inspired by his wide range of expertise, from experiment to numerics and theory. To M. Schuyler Moss, I would like to thank her for making the effort to schedule monthly casual discussions for all of us to share our research. The lessons and advice on compilers Roger has provided has taught me invaluable programming skills and a new way to understand problems. It has been great to have had the opportunity to advise Sehmimul Hoque in a small part. Gebremedhin A. Dagneu and Shiyu Zhou, thank you for teaching me about quantum error correction. The others are not forgotten, they have hugely expanded my horizons with plentiful a discussion.

I would like to thank the current and former members of ions group at the Institute for Quantum Computing (IQC). Prof. Crystal Senko has been a great teacher, her trapped-ion course has imparted onto me deep insight into the light-matter interactions that power the trapped-ion quantum processor. Collin Epstein, Anastasiia Bershanska, Mahmoud Badawy and Mahmood Sabooni has taught me much about the control system driving the trapped-ion quantum processor. Sainath Motlakunta, Nikhil Kothibaskar, Chung-You (Gilbert) Shih, Anthony Vogliano, Jingwen (Monica) Zhu, Yu Ting and others, the work they have put into the development and engineering of the trapped-ion device is inspiring and astounding. Our discussions on the intersection between experiment and theory has been enlightening. This group has convinced me on the great prospects of the trapped-ion device and the future of the field is extremely exciting.

On another front, I extend my thanks to Gregory Allan Dick for his support as the CEO of Open Quantum Design (OQD). The Xanadu catalyst team with Lee O'Riordan, David

Ittah, Mehrdad Malek, Josh Izaac, Paul Wang, Romain Moyard, Joey Carter and company has taught me much about the software development accompanying quantum computers and low level computing infrastructure.

Lastly, I am eternally grateful to my family for their unwavering support.

Dedication

This is dedicated to my family.

Table of Contents

List of Figures	xv
-----------------	----

List of Tables	xviii
----------------	-------

1	Introduction	1
1.1	Many-body systems	2
1.1.1	Ising model	2
1.1.2	Transverse field Ising model (TFIM)	4
1.1.3	Ising lattice gauge theory	5
1.1.4	Classical XY model	6
1.2	Renormalization group (RG)	7
1.2.1	Real-space renormalization group	7
1.2.2	Momentum-space renormalization group	8
1.3	Monte-Carlo methods	9
1.3.1	Markov chain Monte-Carlo (MCMC)	9
1.3.2	Quantum Monte-Carlo (QMC)	10
1.4	Quantum processors	10
1.4.1	Quantum computation	11
1.4.2	Quantum simulation	12
1.4.3	Trapped-ion quantum processor	12

1.4.4	Rydberg atom array quantum processor	19
1.5	Machine Learning	22
1.5.1	Backpropagation	23
1.5.2	Gradient-based optimization	25
1.6	Generative models	26
1.6.1	Energy-based models (EBMs)	27
1.6.2	Flow-based models	30
1.6.3	Autoregressive models	32
1.7	Thesis Outline	35
2	Estimating physical observables from data	38
2.1	Classical shadows	40
2.2	Neural quantum states (NQSs)	41
2.3	Conditional local density RNN (CLDRNN)	44
2.3.1	Local density operators	44
2.3.2	Conditional local density operators	44
2.3.3	Architecture	46
2.4	Long-range transverse field Ising model (TFIM) on the trapped-ion quantum processor	47
2.5	Training procedure for complex RNN wavefunction and CLDRNN	48
2.6	Results	50
2.6.1	Computational cost of machine learning models	53
2.7	Discussion	53
2.8	Conclusion	55
3	Autoregressive model path dependence near Ising criticality	56
3.1	Autoregressive paths	57
3.1.1	Space-filling curves	58

3.1.2	Path properties	59
3.2	Hyperparameters and training procedure	61
3.3	Results	62
3.4	Conclusion	66
4	RydbergGPT: A large language model for Rydberg atom array physics	68
4.1	Rydberg atom array physics	69
4.2	Physical observables	70
4.3	Encoder-decoder transformer architecture	71
4.4	Training transformer models	73
4.5	Inference with trained transformer models	75
4.5.1	Results	76
4.6	Conclusion	80
5	Autoregressive model for continuous variable many-body systems	82
5.1	Architecture	82
5.2	Autoregressive model for the classical XY model	84
5.3	Results	86
5.4	Conclusion	89
6	Fractal transformer: A renormalization group inspired transformer architecture	91
6.1	Architecture	91
6.2	Results	95
6.2.1	Ising model	95
6.2.2	Rydberg system	97
6.2.3	Classical XY Model	98
6.2.4	Ising lattice gauge theory	99
6.3	Conclusion	102

7 Conclusion and outlook	103
References	106

List of Figures

1.1	Diagram of the 2D Ising model on a square lattice.	3
1.2	Diagram of the 2D Ising lattice gauge theory on a square lattice.	5
1.3	Diagram of the 2D XY model on a square lattice.	6
1.4	Illustration of the decimation process in real-space RG.	8
1.5	Markov chain Monte-Carlo.	10
1.6	Typical quantum circuit.	11
1.7	Typical electronic structure of a hydrogen-like ion.	13
1.8	Illustration of pseudo-harmonic ion trap.	14
1.9	Qubit schemes for a trapped-ion quantum processor.	15
1.10	Typical Phonon modes of a trapped-ion quantum processor.	15
1.11	Incoherent operations for a trapped-ion quantum processor.	16
1.12	Two-photon Raman transition for a trapped-ion quantum processor.	18
1.13	Mølmer-Sørensen scheme for a trapped-ion quantum processor.	19
1.14	AC Stark shift.	20
1.15	Rydberg atom array quantum processor.	21
1.16	Artificial neuron.	23
1.17	Multi-layer perceptron.	25
1.18	Backpropagation.	26
1.19	Hopfield model.	27
1.20	Boltzmann machine.	29

1.21	Restricted Boltzmann machine (RBM).	30
1.22	Flow-based model.	31
1.23	Recurrent neural network.	33
1.24	Transformer.	35
2.1	Complex RNN wavefunction.	42
2.2	Circuit diagram for conditional local density.	45
2.3	Conditional local density recurrent neural network (CLDRNN).	46
2.4	Infidelity of NQS state.	50
2.5	Error in the expectation of Pauli observables for the 4-qubit TFIM ground state.	51
2.6	Sampling complexity for TFIM energy.	52
2.7	Estimates for TFIM energy for varying system sizes N .	53
2.8	Training time complexity for complex RNN wavefunction and CLDRNN.	54
3.1	Space-filling curves traversing a 2D square lattice.	58
3.2	Scaling of locality measure $\tilde{C}$ and anisotropy $\mathcal{A}_{xy}$ of autoregressive path with system size N .	61
3.3	Training results for the 2D Ising model near criticality at $\beta = 0.435$ with linear size $L = 8$ for 1D RNN (top) and transformer (bottom).	63
3.4	Evolution of the two-point spin-spin correlations $\mathcal{G}(\Delta x, \Delta y)$ during the training of the transformer for the 2D Ising model near criticality at $\beta = 0.435$ with linear size $L = 8$.	64
3.5	Training of the transformer with the zigzag path on different system sizes.	65
3.6	Training of the transformer with the zigzag path at different temperatures for the 2D Ising model.	65
4.1	Encoding of the system parameters via a graph convolutional neural (GCN).	71
4.2	Encoder-decoder transformer architecture.	72
4.3	Model predictions of multiple observables across the disordered-to-checkerboard phase transition, which is governed by the dimensionless detuning, δ/Ω .	77

4.4	Model predictions of multiple observables as the dimensionless temperature T/Ω is increased.	78
4.5	Extrapolation over lattice size L	79
5.1	Flow-based generator for the transformer.	83
5.2	Training behavior of model $\mathcal{M}_2$ on the XY model.	87
5.3	Trained models $\mathcal{M}_n$ estimates for properties of the XY model.	87
5.4	Performance of model $\mathcal{M}_2$ on multiple sizes of the XY model.	88
5.5	Evaluation of a model $\mathcal{M}_4$ trained on multiple sizes of the XY model. . . .	88
5.6	Transfer learning of model $\mathcal{M}_2$ for size $L = 16$ XY model.	89
6.1	Neural network for the decimation of spins in the fractal transformer. . . .	93
6.2	Schematic for the hierarchy of transformer calls in the fractal transformer. .	93
6.3	Fractal transformer estimates for the properties of the 2D Ising model across the phase transition.	96
6.4	Collapse in the magnetic susceptibility for the 2D Ising model estimated by the fractal transformer.	96
6.5	Fractal transformer estimates for the properties of the 2D Ryberg Hamiltonian across the phase transition.	98
6.6	Evaluating the continuous fractal transformer applied to the 2D XY model.	99
6.7	The 3D Ising lattice gauge theory on a cubic lattice and the tetrahedral grouping of spins.	100
6.8	Fractal transformer prediction for the Wilson loops of the Ising lattice gauge theory.	101

List of Tables

1.1	Critical exponents for Ising models of various dimensions.	4
1.2	Common activation functions used in neural networks.	24
2.1	The hyperparameters for the training of the complex RNN wavefunction and CLDRNN on the long-range TFIM.	49
3.1	Properties of 2D space-filling curves.	60
3.2	Hyperparameters for the training of 1D RNN and transformer on the 2D Ising model.	62
4.1	Number of tokens, in units of 10^8 , contained in our dataset for various sizes L	74
4.2	The hyperparameters for the transformer encoder-decoder architecture and optimizers.	74
4.3	Details on the training of multiple RydbergGPT models.	75
5.1	Hyperparameters of the transformer, with flow generator, and the training procedure.	85
5.2	Details on the training dataset of multiple transformer models for different sizes of the XY model.	86
6.1	Hyperparameters of the fractal transformer architecture and training.	95
6.2	Details on the dataset used to train multiple fractal transformer models.	95
6.3	Hyperparameters of the fractal transformer architecture for the 3D Ising lattice gauge theory.	101

Chapter 1

Introduction

The study of many-body physics brings about advances in quantum information processing, material science, machine learning and our fundamental understanding of the universe. In the field of quantum information processing, many-body physics exhibits a close relation with error correction protocols, such as the toric code and other topological codes [3, 4]. For material science, much is still unknown about the mechanism that leads to the emergence of superconductivity in high temperature superconductors, further studies of many-body physics models aimed to explain this phenomenon naturally power future efforts to search for or design superconducting materials [5–7]. In recent years, efforts have been made to gain insight into the power and behavior of machine learning via reframing these systems as statistical field theories [8–10]. Our best models for the universe consist of quantum field theories, these systems have a bilateral correspondence with many-body systems, development of analysis methods are widely shared between these fields.

On the other hand, the accelerating development of quantum processors is exciting for the advancement of many-body physics. Quantum simulation of many-body systems at scale [11–16] could be realized in the near future, contributing to the repertoire of tools physicist could employ in the investigations of such systems.

Another area that has experienced multiple breakthroughs over the past few decades is machine learning. Machine learning has demonstrated extraordinary capability to recognize patterns, generalize and scale. In particular, autoregressive generative models, such as the transformer, have shown remarkable performance in natural language processing task [17–19]. This has led to the commodification of these large language models (LLMs) in today’s world, famous examples include ChatGPT [20], Gemini [21], DeepSeek [22], etc. Furthermore, machine learning has proven useful in a wide variety of fields in physics, such

as, many-body physics [23–25], quantum control [26, 27], quantum information processing [28–30], high energy physics [31], fluid dynamics [32], cosmology [33, 34] etc.

In this thesis, we explore the use of autoregressive generative models, particularly the transformer, to study typical many-body physics models of the classical and quantum variant. The quantum many-body models investigated are heavily influenced by the capabilities of currently available atom-based quantum simulators. We investigate the influence on the behavior of autoregressive generative models associated with the choice of autoregressive path. Additionally, we probe the generalization power of these autoregressive generative models in multiple axes. Finally, we propose many-body physics inspired architectural modifications to improve the generality of such models.

1.1 Many-body systems

1.1.1 Ising model

The Ising model is a classic model in the field of many-body physics [35]. The system describes the emergence of magnetism in a group of spins with the following expression for the energy:

$$E = \pm \sum_{\langle i,j \rangle} x_i x_j, \quad x_i \in \{\pm 1\}. \quad (1.1)$$

Each spin experiences an interaction with its nearest neighbors. Based on the sign of the interaction, the spins would be preferably aligned or anti-aligned, resulting in ferromagnetism and anti-ferromagnetism respectively. Additionally, the Ising model exhibits a $\mathbb{Z}_2$ symmetry, i.e. flipping every spin leaves the energy unchanged. Due to the $\mathbb{Z}_2$ symmetry, the ground state of the system has a degeneracy of 2. At low temperatures, the model undergoes spontaneous symmetry breaking and ends up in one of the two ground states. The Ising model can be applied to lattices of arbitrary dimensions or even arbitrary graphs.

In one dimension, the spins form a chain, the chain can either be open or closed. An open chain has the spins at the two ends of the chains unconnected, this is called an open boundary condition (OBC). Whereas, the closed chain has the spins at the two ends connected, this is called a periodic boundary condition (PBC). The same property applies to arbitrary dimensional Ising models that live on a lattice, they can have OBCs or PBCs. The one-dimensional (1D) Ising model was solved by Ising [35], he showed that the model does not exhibit a phase transition.

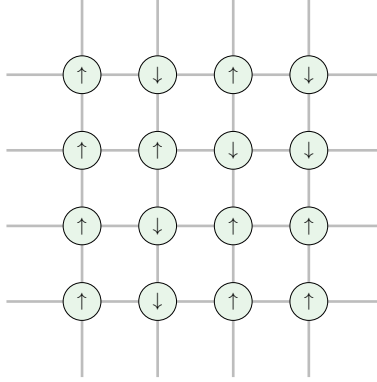


Figure 1.1: **Diagram of the 2D Ising model on a square lattice.** The spins are discrete 2-level systems and are placed on the vertices of the lattice.

For the two-dimensions and higher, Peirels argument proves that the system exhibits a phase transition [36] between an ordered phase and a disordered phase, where the spins are aligned in an ordered way or the spins are orientated randomly respectively. As the lowest dimensional Ising model with a phase transition, the lower critical dimension of the Ising model is $d_c^{\text{lower}} = 2$. There exist a duality between the high and low temperature regimes of the 2D Ising model, this duality is known as the Kramers-Wannier duality [37]. The Kramers-Wannier duality was the first to theoretically determine the critical point where the phase transition occurs, at temperature,

$$T_c = 2 / \ln(1 + \sqrt{2}). \quad (1.2)$$

The 2D Ising model was analytically solved by Onsager [38] and the solution is known as Onsager's solution. This phase transition in the 2D Ising model is the namesake of the 2D Ising universality class of phase transition, of which many other phase transitions belong to.

There is no known analytic solution to the 3-dimensional (3D) Ising model on the cubic lattice. One method used to study this system is Monte-Carlo simulations (see section 1.3), where the equilibrium configuration of the system is sampled. The Monte-Carlo estimate of the critical temperature $T_c \approx 4.511$ [39, 40]. The critical point, where the phase transition occurs, in the 3D Ising model on a cubic lattice is known as the Wilson-Fisher fixed point. This fixed point was determined via renormalization group (RG) techniques (see section 1.2) [41]. This phase transition is the namesake of the 3D Ising universality class.

The infinite dimension Ising model corresponds to an all to all interaction between the spins and can be solved exactly with mean-field theory. In mean-field theory, the spins

	Dimension d	2	3	≥ 4
Critical exponents	ν	1	0.63	$\frac{1}{2}$
	α	0	0.11	0
	γ	$\frac{7}{4}$	1.24	1

Table 1.1: **Critical exponents for Ising models of various dimensions.** Here, ν, α and γ are the critical exponents of the correlation length, specific heat capacity and susceptibility respectively. The values for the critical exponents can be derived analytically for 2D and $d \geq 4$ systems. For 3D systems, the critical exponent is taken from [39, 42].

are approximated to be interacting with a global field instead of between one another simplifying the system. Another method of solving the infinite dimensional Ising model is using Landau theory [43, 44]. In Landau theory, the microscopic properties of the system are ignored and instead the theory focuses on the macroscopic properties. The Landau theory when applied to Ising models of greater than $d = 4$ dimensions demonstrate that they abide by the mean-field solution. Consequently, $d_c^{\text{upper}} = 4$ is the upper critical dimension of the Ising model.

The critical exponents for the phase transitions exhibited by the d -dimensional Ising models, explored above, are collated in table 1.1.

1.1.2 Transverse field Ising model (TFIM)

The transverse field Ising model (TFIM) is the quantum generalization of the classical Ising model:

$$\hat{H} = \pm \sum_{\langle i,j \rangle} \hat{\sigma}_i^x \hat{\sigma}_j^x + h \sum_i \hat{\sigma}_i^z. \quad (1.3)$$

Here, h is the transverse field strength. Like its classical counterpart for thermal phase transitions, the transverse field Ising model is a foundational model for studying quantum phase transition, where the field h is varied.

Instead of the nearest neighbor interaction, if we instead replace it with a power law interaction between all spins, we obtain the long-range TFIM, extensively studied in recent years [45–48]. The long-range TFIM is of particular interest as they are natively simulatable on a variety of quantum processors [11, 13, 49–51]. These models are highly interesting

as they are naturally frustrated. Furthermore, the long-range property relates to faster information transfer throughout the system useful for efficient quantum signalling [52].

1.1.3 Ising lattice gauge theory

The Ising lattice gauge theory is a generalization of the Ising model where, instead of the spins placed on the 0-dimensional vertices and the interaction on the 1-dimensional (1D) edges, the spins are placed on the 1D edges and the interactions on the 2-dimensional plaquettes. The plaquette is a four-body interaction involving the spins bordering the plaquette resulting in the following energy:

$$E = \pm \sum_{\square} x_i x_j x_k x_l, \quad x_i \in \{\pm 1\}, \quad (1.4)$$

where, $\sum_{\square}$ represents a sum over all possible plaquettes, an example of which is shown in fig. 1.2.

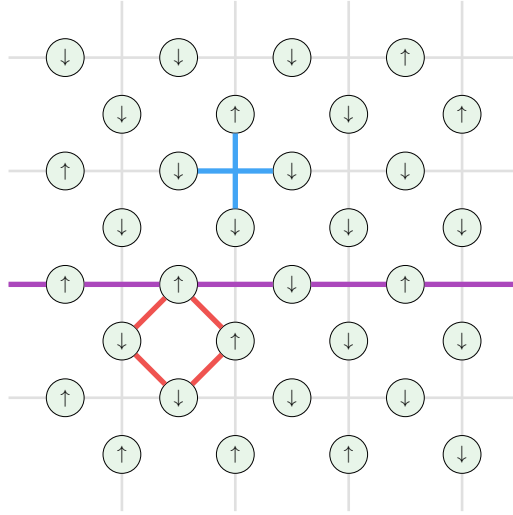


Figure 1.2: **Diagram of the 2D Ising lattice gauge theory on a square lattice.** The spins are discrete 2-level systems placed on the edges of the lattice. The red square represents a plaquette operator, the blue cross a star operator and the purple line is a Wilson loop.

Similar to the Ising model, the Ising lattice gauge theory has a $\mathbb{Z}_2$ symmetry. However, the Ising lattice gauge theory has a gauge degree of freedom, i.e. there is a local symmetry that

leaves the energy invariant. This gauge degree of freedom makes the Ising lattice gauge theory related to other gauge theories that arise in quantum chromodynamics [53, 54] and quantum electrodynamics, thus motivating its study. The gauge degree of freedom on the Ising lattice gauge theory corresponds to the star flip (shown in fig. 1.2 as a blue cross).

By Peirels argument [36], the Ising lattice gauge theory has a lower critical dimension $d_c^{\text{lower}} = 3$. The 3D Ising lattice gauge theory on a cubic lattice can be related to the 3D Ising model via Wegner duality, a generalization of Kramers-Wannier duality [55]. Consequently, the phase transition for the 3D Ising lattice gauge theory belongs to the 3D Ising universality class with the same critical exponents as shown in table 1.1. The critical temperature of the 3D Ising lattice gauge theory is $T_c \approx 1.31$ [56]. It is proven that the 3D Ising lattice gauge theory does not have a local order parameter [57]. The transition is studied via the examination of the Wilson loops, the product of spins along a loop, an example of which is the purple line in fig. 1.2 .

The Ising lattice gauge theory can be further generalized for DoFs on the s -dimensional hyperface and the interactions on an $s + 1$ dimensional hypercube [55].

1.1.4 Classical XY model

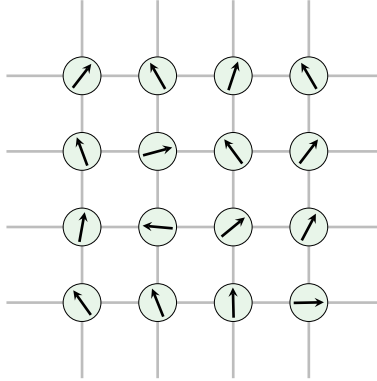


Figure 1.3: **Diagram of the 2D XY model on a square lattice.** The rotors are continuous DoFs, parameterized by a single angle, placed on the vertices of the lattice.

Instead of generalizing the dimension of the hyperface the spins live on, we consider replacing the discrete spins of the Ising model with a 2D rotor at each site:

$$E = \pm \sum_{\langle i,j \rangle} \cos(\theta_i - \theta_j), \quad \theta_i \in [-\pi, \pi), \quad (1.5)$$

where, θ_i is the angle that indicates the direction the rotor- i is pointing in. This forms the XY model, a pivotal model that forms our understanding of superfluidity. The system has a $U(1)$ continuous global symmetry, i.e. a uniform rotation of all the rotors does not modify the energy.

The XY model experiences the well known Berezinskii-Kosterlitz-Thouless (BKT) transition [58, 59] between disordered phase and a quasi long-range ordered phase at critical temperature $T_c = 0.89273$ [60]. Due to vortex generation in the XY model, the BKT transition exhibits an algebraic decay of correlations near criticality instead of the typical exponential decay.

1.2 Renormalization group (RG)

Renormalization group (RG) is a method with great success in the study of many-body systems. In particular, RG provides a mechanism for studying models as they are scaled in size, up to the thermodynamic limit. RG capabilities arise from the introduction of a scale transform, used to grow the system while ignoring asymptotically irrelevant aspects of the model. The process is conditioned on the self-similarity of the model at multiple scales in the representation RG is performed in. The method gives us insight into the behavior of many-body systems by providing a way to theoretically obtain estimates for the critical properties of the system as it undergoes a phase transition.

1.2.1 Real-space renormalization group

Real-space RG, also known as Kadanoff RG was introduced by Kadanoff to solve the 2D and 3D Ising models on the square and cubic lattice respectively [61, 62]. The procedure involves decimating a local patch of spins into a block-spin with magnetization equal to the mean magnetization of the local patch. For the Ising model, the process of doing so does not change the structure of the model, i.e. the model is self-similar under this decimation procedure. Consequently, the decimation process can be iteratively applied to the system. During the iterative decimations, the parameters of the model are modified. RG involves studying the system of equations that govern the evolution of the model parameters as a single decimation step is performed, these are the RG equations that describe the RG flow. As the model is decimated, the parameters flow into particular fixed points. These fixed points each correspond to either phases or critical regions of the model. At criticality, models typically experience a divergence in the correlation length. Due to this divergence,

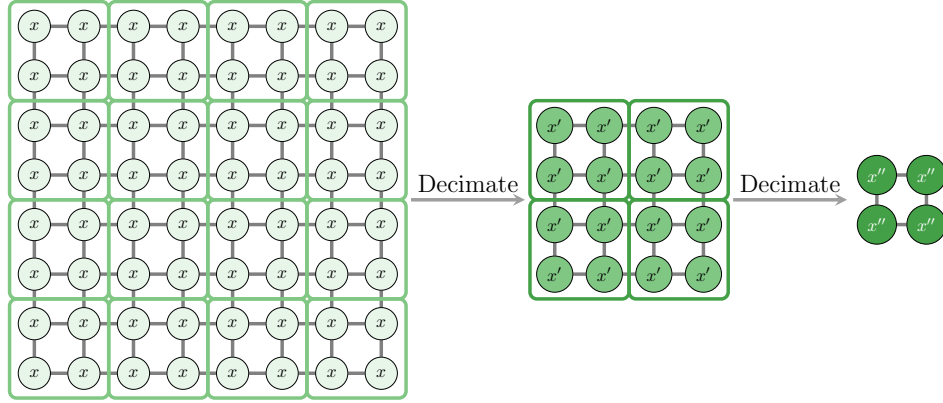


Figure 1.4: **Illustration of the decimation process in real-space RG.** As each decimation step is performed we obtain a higher order block-spin $x^{(\mathcal{O})}$.

there exist no length scale in the problem, resulting in a fully self-similar model, i.e. model parameters are not modified by the RG flow. However, these fixed points are unstable, perturbations near criticality drive the system into stable fixed points that correspond to different phases. By determining the parameter regimes in the phase diagram that are attracted to each fixed point, under RG flow, we determine the phase boundaries where phase transitions occur. Furthermore, the equations describe the properties near criticality, i.e. the critical exponents, allowing us to determine the universality class of the phase transition.

In the real-space basis, we are limited in the models where RG applies. With different lattice geometries and different models, the system is no longer self-similar under the real-space decimation procedure, resulting in the invalidity of RG.

1.2.2 Momentum-space renormalization group

Momentum-space RG, also known as Wilsonian RG, expands on the RG process by considering a different basis [41, 63–65]. Instead of performing RG in real-space, suppose a Fourier transform is first applied mapping the system into momentum space. RG is then performed on the model expressed in the momentum basis. The key realization is that in momentum space, high wavenumber components correspond to short-range effects, as such, implementing a ultraviolet (UV) cutoff is analogous to coarse-graining. After the UV cutoff, the system is rescaled to the original scale, in doing so the system parameters would be updated with new values. Similar to before, the updates are governed by a set of RG

equations. Solving these RG equations informs us about the phases and phase transitions that exist in the many-body model.

1.3 Monte-Carlo methods

1.3.1 Markov chain Monte-Carlo (MCMC)

The purpose of MCMC is to sample from a target joint probability distribution. In this manuscript, we are interested in sampling from a Boltzmann distribution for an energy function describing a many-body system,

$$\mathbf{x} \sim P(\mathbf{X}) = \text{Boltzmann}[E](\mathbf{X}), \quad (1.6)$$

$$\text{Boltzmann}[E](\mathbf{X} = \mathbf{x}) = \frac{e^{-\beta E(\mathbf{x})}}{\mathcal{Z}}, \quad \mathcal{Z} = \sum_{\mathbf{x}} e^{-\beta E(\mathbf{x})}. \quad (1.7)$$

The Metropolis-Hastings algorithm [66, 67] used to perform the sampling is as follows:

1. Start with sampling the configuration $\mathbf{x}$ from an initial distribution Q .

$$\mathbf{x}_0 \sim Q(\mathbf{X}). \quad (1.8)$$

2. Propose an update to the the configuration and accept the proposal based on transition probability:

$$T[E](\mathbf{x}_t \rightarrow \mathbf{x}_{t+1}) = \min \left(1, \frac{P(\mathbf{x}_{t+1})}{P(\mathbf{x}_t)} \right). \quad (1.9)$$

3. repeat step (2) until the Markov chain equilibrates.
4. repeat step (2) to generate samples.

In step (4) of the Metropolis-Hastings algorithm, we generate samples by iteratively applying Monte-Carlo steps. Due to the Markovian process, the samples are autocorrelated in time. By pruning the samples generated, i.e. only collecting samples separated by the autocorrelation time, we are able to obtain approximately independent and identically distributed (IID) samples with MCMC.

Another issue with MCMC is that the Markov chain could end up frozen in a particular configuration for low-temperature systems. Annealing [68] and parallel tempering [69, 70] are two methods that address this problem.

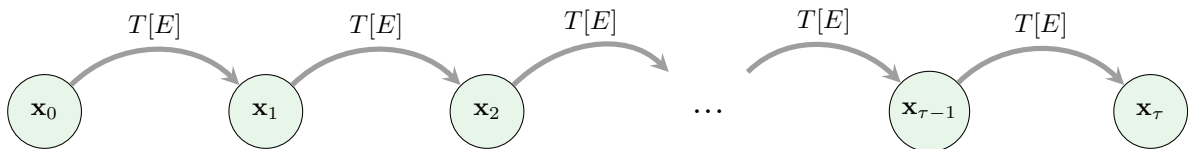


Figure 1.5: **Markov chain Monte-Carlo.** The state at a time step x_t , depends only on the state immediately preceding it x_{t-1} . At each step, a proposed move is made with an acceptance probability that depends on the energy of the initial and proposed states $T[E]$.

1.3.2 Quantum Monte-Carlo (QMC)

Quantum Monte-Carlo (QMC) consist of a family of algorithms that enable the simulation of many-body quantum systems. The flavours of QMC include:

- Green’s function Monte-Carlo [71]
- Path integral Monte-Carlo [72, 73]
- World line Monte-Carlo [74]
- Stochastic series expansion (SSE) [75–77]

In path integral and world line Monte-Carlo, the Suzuki-Trotter decomposition [78, 79] is applied to the partition function. Whereas, in stochastic series expansion, the partition functions is decomposed via Taylor expansion. These methods introduce a mapping between d -dimensional quantum systems to a $d + 1$ dimensional classical system [80] prime for simulation with MCMC introduced in the previous section.

A well-known set of problems with QMC are the sign problems associated with fermionic systems [81, 82] and frustrated systems [83, 84].

1.4 Quantum processors

In the recent decade, the development of quantum processors has accelerated with many efforts to prove quantum advantage over conventional classical computing [85, 86]. A quantum processor is a well-behaved controllable quantum mechanical system. These processors are expected to revolutionize multiple fields in the coming decades, including metrology [87], material science [88], medicine [89], etc.

1.4.1 Quantum computation

A typical framework to view programs ran on a quantum computer is based on discrete computational blocks, i.e. gates. In this framework, the gates are composed with measurements to form circuits, as shown in fig. 1.6.

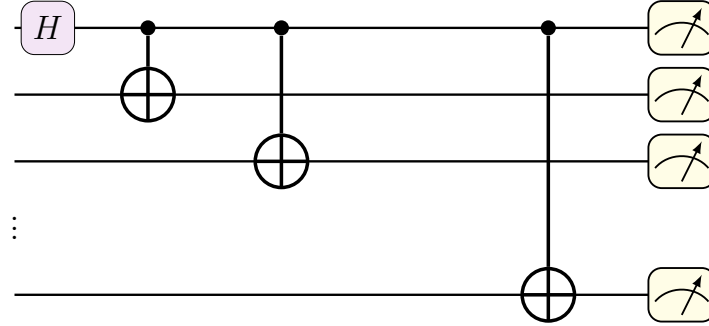


Figure 1.6: **Typical quantum circuit.** The circuit in the diagram is the standard preparation procedure for the Greenberger-Horne-Zeilinger (GHZ) state.

The DiVincenzo's criterion [90] outline the requirement for quantum computers:

1. A physical system that is capable of encoding a qubit, i.e. a quantum two-level system, and this qubit is well insulated from its surroundings.
2. A protocol to prepare the qubits in one of the states with high fidelity.
3. The qubit demonstrates a long coherence time.
4. A universal gate set for manipulating the qubits.
5. A protocol for measuring the current state of the qubits.

A universal gate set consist a set of gates that allow for arbitrary rotations of a single qubit and an entangling gate between pairs of qubits. Together, the universal gate set is able to perform arbitrary computation on the state.

Many architectures of quantum computers have been proposed and are in active development, such as trapped-ions, Rydberg atom arrays, superconducting qubits, photonics, etc.

1.4.2 Quantum simulation

In contrast to quantum computation, for quantum simulation we are looking to reproduce the Hamiltonian of a physical system in our quantum processor. Each quantum processor architecture has a set of native interactions that give rise to the gates of the system. In particular architectures, the native interactions permit a wider range of possible interactions via finer control over the processor parameters [27, 91–93]. We use the flexibility in the native interactions to mimic the physical system of interest, such as molecules, materials or many-body systems. Under these interactions, the quantum processor behaves as a well-controlled and probeable analogue enabling in depth studies of the system of interest. Another approach for quantum simulation would be to decompose the Hamiltonian of the physical system into gates or simpler Hamiltonians [94].

1.4.3 Trapped-ion quantum processor

The trapped-ion quantum processor is comprised of a set of ions suspended in vacuum using electrostatic and electrodynamic fields. The system of the ions are then manipulated with electromagnetic fields and lasers to perform quantum operations.

Ion selection

Typically, hydrogen-like ion species are used due to their simple energy level structure demonstrated in fig. 1.7. These ion are not only alkali-like, i.e. they have one unfilled orbital populated by exactly one electron, they also have simple nuclear structure, i.e. having a nuclear spin of $1/2$. Ions that satisfy this criteria include $^{133}\text{Ba}^+$, $^{171}\text{Yb}^+$.

Another consideration when picking the ion species is the transition energies between the energy levels. The ideal ion would have transition energies commensurate with the frequencies of available high-powered lasers. This allows the optical system to be comprised of commercially available components that are easily procured.

Electrodynamic Trap

Ions experience Coulomb forces when interacting with electric fields due to their charged nature, as such we utilized these fields to confine the ions. The Earnshaw’s theorem states that it is impossible to trap a charged particle with only an electrostatic fields as this would require a violation of Laplace’s equation. Therefore, a combination of electrostatic

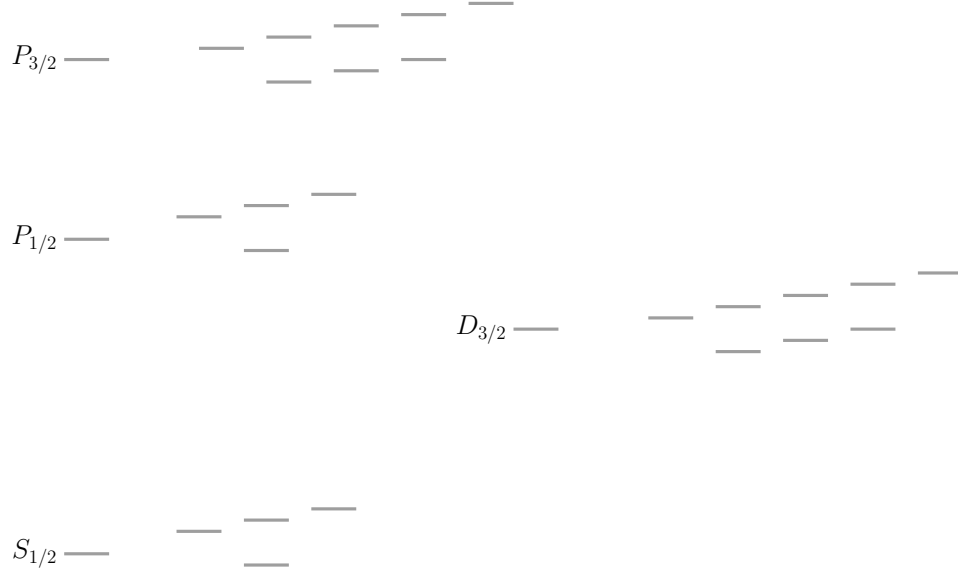


Figure 1.7: **Typical electronic structure of a hydrogen-like ion.**

and electrodynamic fields are used to trap the ions. The electrostatic fields are typically generated by apply a direct current (DC) on an electrode, whereas the electrodynamic fields are generated by applying a radio-frequency (RF) current on the electrode. After time-averaging the Coulomb forces experienced by the ions, the ions are approximately confined by a harmonic pseudo-potential, in all three directions, near the vicinity of the trap. This procedure of trapping the ions is known as the Paul trap [95]. The total potential experienced by the ions take the following form in the vicinity of the trap:

$$\Phi^{\text{total}} = \Phi^{\text{Coulomb}} + \Phi^{\text{Paul}}, \quad (1.10)$$

$$\Phi^{\text{Coulomb}} = \sum_{i < j} \frac{q_i q_j}{4\pi\epsilon_0 \|\mathbf{r}_i - \mathbf{r}_j\|}, \quad (1.11)$$

$$\Phi^{\text{Paul}} \approx \sum_i \frac{M}{2} (\omega_x^2 x_i^2 + \omega_y^2 y_i^2 + \omega_z^2 z_i^2). \quad (1.12)$$

Here, Φ^{total} is the total potential experienced by the ions, Φ^{Coulomb} is the Coulomb potential between the ions, Φ^{Paul} is the time-averaged harmonic pseudo-potential generated by the DC and RF electrodes. The position, mass and charge of ion- i are $\mathbf{r}_i$, M and q_i respectively. ϵ_0 is the permittivity of free space, a universal constant.

Typically the ions are trapped as a 1D Coulomb crystal by setting the axial trapping strength to be much less than the transverse trapping strengths, i.e. $\omega_z \ll \omega_{x/y}$ [96, 97].

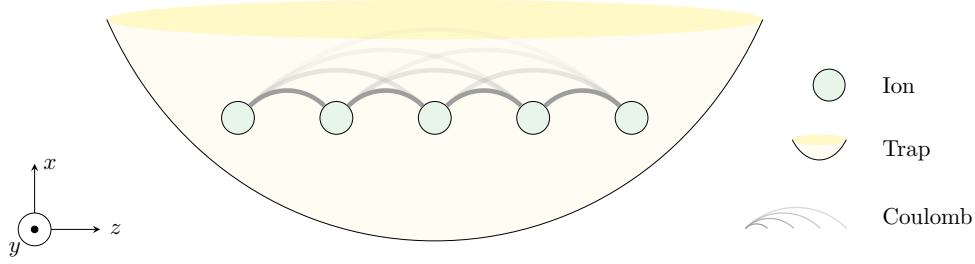


Figure 1.8: **Illustration of pseudo-harmonic ion trap.** The ions are trapped in a chain given the trapping strength obey $\omega_z \ll \omega_{x/y}$. Due to charge of the ions, they experience long-range Coulomb interactions.

Quantum degrees of freedom

As previously mentioned, ion species are decided upon based on their electronic structure. These electronic states are used to encode quantum information. Typically, quantum experiments are designed based on a two-level system, forming a qubit. As such we encode a qubit into the electronic states of the ion. There are multiple schemes for this procedure:

- Hyperfine qubits are formed by encoding the qubit states into the two of the hyperfine ground-states of the ion [98].
- Optical qubits are formed by encoding the qubit states between a ground-state of the ion and a meta-stable excited state of the ion [99]. The energy difference between these two states is typically in the optical regime.

In both of the above schemes, the electronic states of ions chosen to encode the qubit are long-lived, satisfying condition (1) and (3) of DiVincenzo's criterion.

Another quantum degree of freedom in the trapped-ion system arises due to the combination of Coulomb repulsion between ions and the trap potential experienced by the ions. When the ions are trapped, they form a Coulomb crystal. The Coulomb crystal behaves like a solid, as such they exhibit collective behavior. The collective behavior of the crystal corresponds to vibration that propagate within the crystal, i.e. phonons. At low temperatures, these phonons form a bosonic quantum degree of freedom of the system.

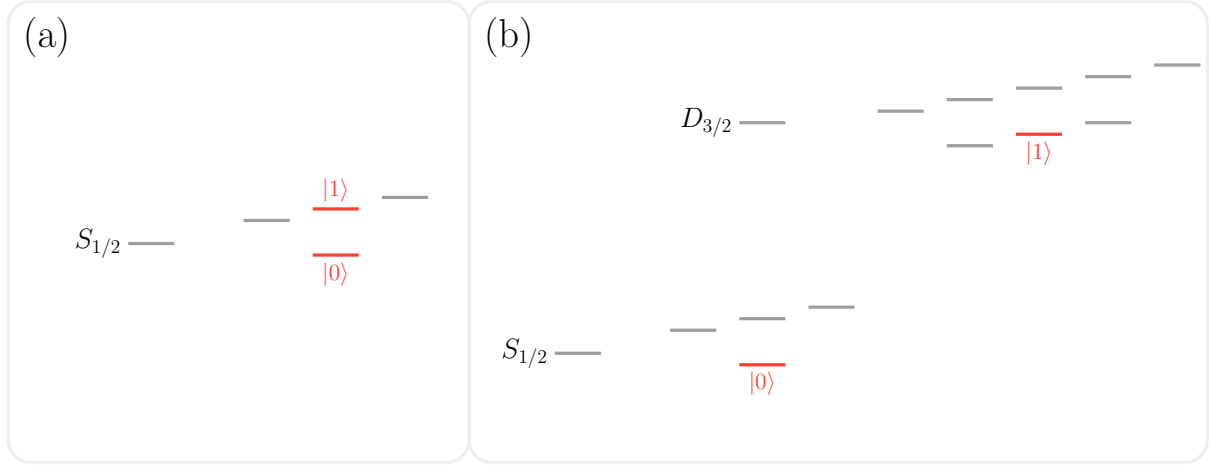


Figure 1.9: **Qubit schemes for a trapped-ion quantum processor.** (a) Hyperfine qubit and (b) optical qubit.

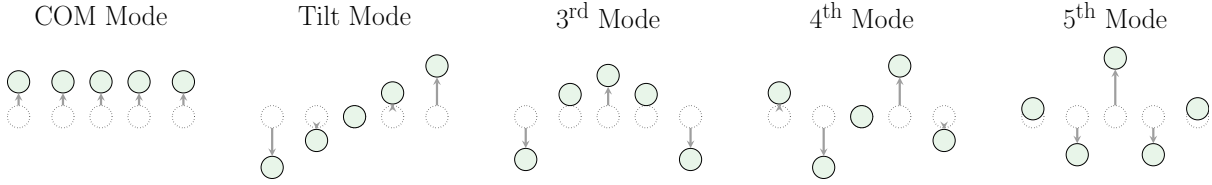


Figure 1.10: **Typical Phonon modes of a trapped-ion quantum processor.** Here, we show the transverse eigenvectors for a 5 ion system.

Light-matter interactions

So far, I have described the properties of the trapped-ion system, in the following I will describe how electromagnetic fields are used to manipulate the trapped-ion system to perform quantum experiments.

To prepare the system for a quantum experiment, the phonon modes of the ion crystal are first cooled to low temperatures. This is typically done in multiple stages to attain the lowest temperatures. The first stage is Doppler cooling, a classical process, where the Doppler effect is leveraged to cool the ions motion with a laser [100]. The frequency of the laser is tuned such that when the ion is moving towards the source of the laser beam, the laser is resonant with a particular electronic transition of the ion to a short-lived excited

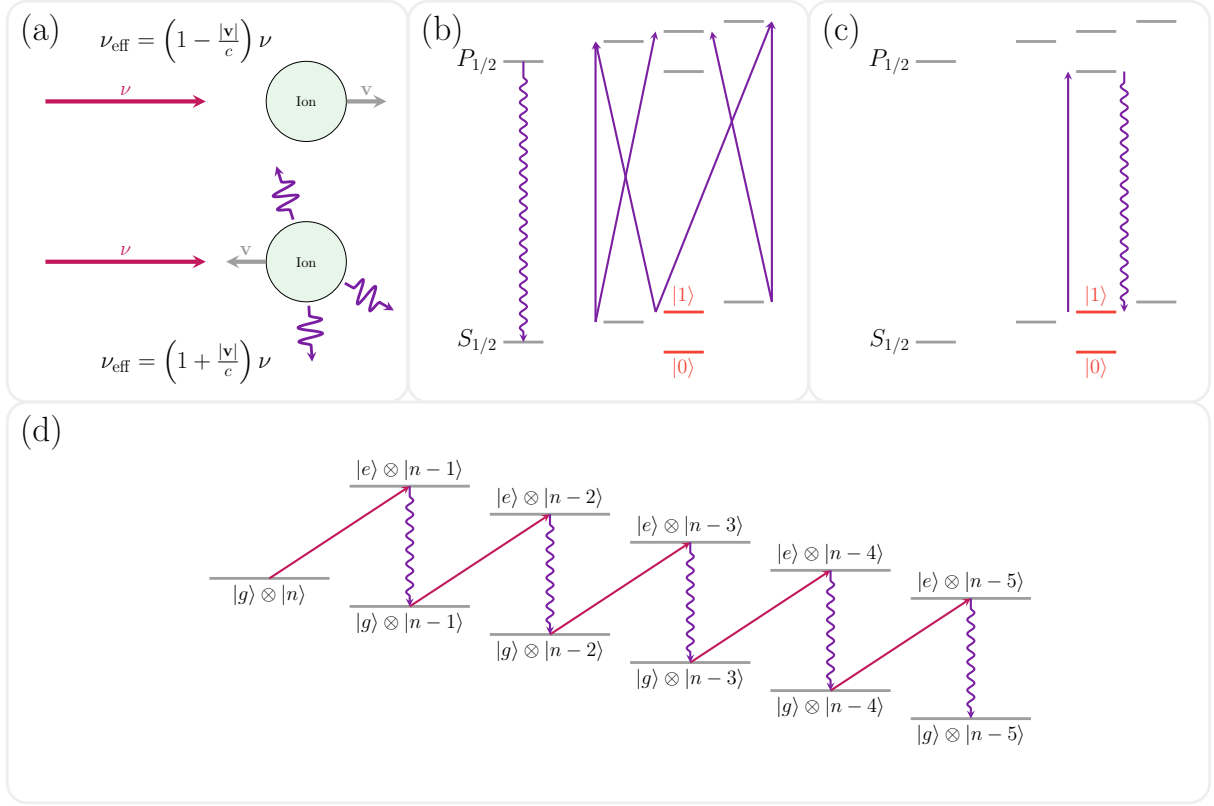


Figure 1.11: **Incoherent operations for a trapped-ion quantum processor.** (a) Doppler cooling, (b) optical pumping, (c) fluorescent measurement and (d) resolved side-band cooling.

state. This results in the ion conditionally absorbing a photon from the laser. As the photon has some momentum that is opposite in direction with the movement of the ion, by the conservation of the momentum, the magnitude of the momentum possessed by the ion is reduced accordingly. Due to the short-lived nature of the excited state. The ion quickly decays back into the ground-state while emitting a photon. The emitted photon propagates in a random direction and the ion experiences a corresponding kick. However, due to the randomness in the direction of the kick, on the whole, the ions momentum is reduced. This reduces the motion of the ions, hence, the ion is cooled. Due to the random kicks experienced by the ion, the motion of the ions never reach zero, as such there is a limit to the temperatures reachable via Doppler cooling. This temperature is known as

the Doppler temperature and takes the form $T_{\text{Doppler}} = \hbar\gamma/(2k_B)$, where $\hbar$ is the reduced Planck's constant, γ is the linewidth of the transition and k_B is Boltzmann's constant.

To cool the ions below the Doppler limit, resolved sideband cooling [101, 102], electromagnetically induced transparency cooling [103], etc. can be employed. Resolved sideband cooling involves driving a resonant red-sideband pulse of a particular transition and phonon mode (see fig. 1.11(d)). After which the ion is allowed to decay back into the ground-state. The above procedure is repeated allowing for the iterative extraction of energy from the phonon modes, reducing the population of the phonon mode and thus the temperature.

After cooling the phonon modes of the ion crystal, the ions are prepared in one of the qubit states via the optical pumping procedure [104]. Optical pumping involves driving a resonant transition between the non-target qubit state and a short-lived excited state (see fig. 1.11(b)). The short-lived excited state probabilistically decays into the target qubit state. If the ion decays into some other state, those states are driven back to the qubit subspace with additional repump lasers. Overall, the population in the target qubit states increases and asymptotically saturates to 1. This allows the qubit to be initialized in one of the qubit state with high fidelity satisfying (2) of DiVincenzo's criterion.

Measurements in the trapped-ion platform are performed via the driving of a fluorescent transition between one of the qubit states and a short-lived excited state forbidden from transitioning into the other qubit state [105]. The short-lived excited state either decays back into the initial state or to some other state outside the qubit space. Repump lasers are used to drive the outside states back to the initial state, ensuring a closed cycling transition between the initial state and the excited state. As the protocol is performed, the ion fluoresces producing photons that are collected by an imaging system. The presence of fluorescence indicate that the ion is in the target qubit state whereas lack of fluorescence indicate that the ion is in the non-target qubit state. This protocol addresses (5) of DiVincenzo's criterion.

Once the qubit is prepared, single qubit operations are performed by driving the qubit transition with a direct beam or a two-photon Raman transition, this equates to the following Hamiltonian:

$$\hat{H} = \frac{\Omega}{2} (e^{i\phi}\hat{\sigma}_+ + e^{-i\phi}\hat{\sigma}_-). \quad (1.13)$$

Here, Ω is the Rabi frequency and ϕ is the effective phase of the driving beam and affects the axis of the one-qubit rotation. For a hyperfine qubit, the direct beam would typically be of a microwave frequency. This limits the ability to focus the beam for individual addressing, as such the two-photon Raman transition [104] is instead used for general one-qubit operations on a trapped-ion quantum computer. The two-photon Raman transition

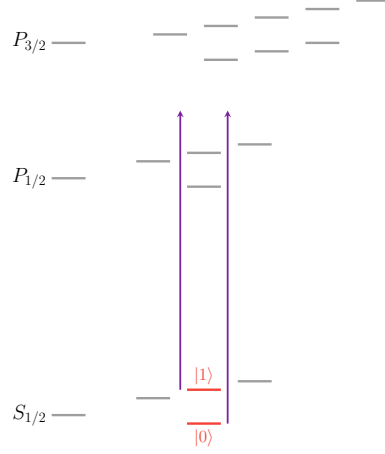


Figure 1.12: **Two-photon Raman transition for a trapped-ion quantum processor.** The transition is driven by a pair of frequencies detuned specifically such that they are off-resonant with the excited states and the difference in frequencies is the qubit frequency.

couples the qubit manifold with an excited state off-resonantly (see fig. 1.12(a)). When the two Raman beams are far-detuned, the excited state is only virtually excited, i.e. is not populated significantly. With this procedure, the frequency of the Raman beams are detuned by the qubit frequency and each beam is of a frequency near the frequency of the excited state transition, typically in the optical regime. As such, the two-photon Raman transition enables individual addressing in the trapped-ion system.

The entangling gates of the trapped-ion system is typically performed with Mølmer-Sørensen scheme [104, 106]. By off-resonantly driving a pair of blue and red sidebands, the ions are driven with a spin-dependent force. The spin-dependent force imparts momentum into the ion crystal, exciting the phonon modes of the system resulting in a coupling between the qubit degrees of freedom and the phonon degrees of freedom. With sufficient detuning on the sidebands, the phonon modes are only virtually excited and can be adiabatically eliminated, resulting in a pure interaction in the qubit space. The effective Hamiltonian experienced by the ions is as follows:

$$\hat{H} = \sum_{i < j} J_{ij} \hat{\sigma}_i^x \hat{\sigma}_j^x \quad (1.14)$$

where J_{ij} is the interaction strength between ion- i and ion- j . The interaction strength can be controlled by adjusting the parameters of the beams driving the blue and red sidebands.

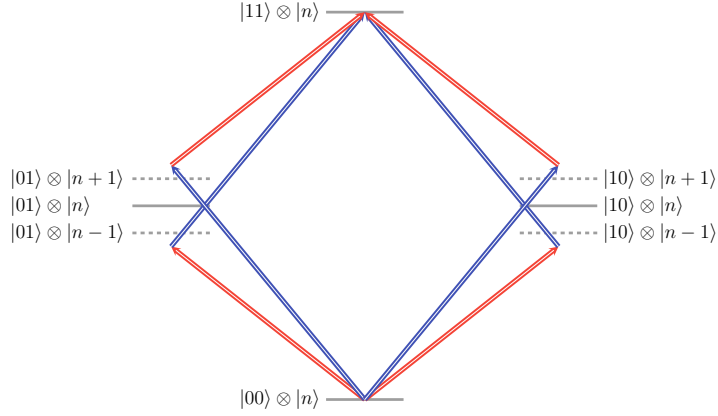


Figure 1.13: **Mølmer-Sørensen scheme for a trapped-ion quantum processor.** The protocol consist of driving the red and blue sidebands. Typically, for hyperfine qubits these sidebands are driven by a pair of counter-propagating beams via the Raman transition.

Another procedure to decouple the phonon modes in the Mølmer-Sørensen scheme is via the precise pulse shaping of the beams such that the phonons are uncoupled from the spin at specific times. This allows for faster two-qubit gates as there is no restrictions on the population of the phonon states during the evolution.

The combination of the two-photon Raman transition and the Mølmer-Sørensen scheme fulfills (4) of DiVincenzo's criterion. In addition to quantum computation, one can use the trapped-ion system for quantum simulation as well, e.g. by making use of the procedures introduced above one can engineer the TFIM on the trapped-ion system [13, 49, 50].

1.4.4 Rydberg atom array quantum processor

The Rydberg atom array is another platform for quantum processing. It makes use of an optical tweezer array to suspend neutral atoms. The internal state of the atoms are then manipulated with electromagnetic fields and lasers. The Rydberg atom array bears remarkable similarities to the trapped-ion quantum processor as they are both based on the light-matter interaction in atomic systems.

Ion selection

The choice of atom in the Rydberg atom array depends on the frequencies of the electronic state transitions. Ideally, commercially available lasers should be able to drive these

transitions. Typical choices include ^{87}Rb , ^{87}Sr , ^{133}Cs .

Optical tweezer array

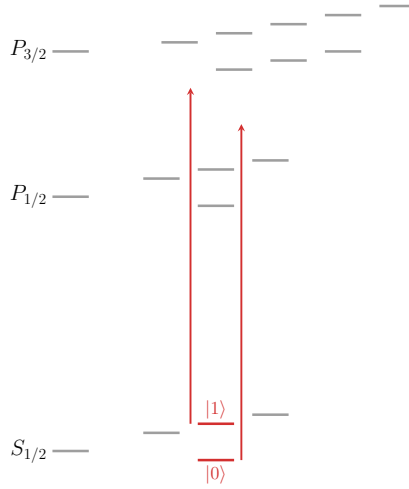


Figure 1.14: **AC Stark shift.** The red arrows represent the same laser when interacting with different electronic levels.

Optical tweezers, as shown in fig. 1.15, make use of the alternating current (AC) Stark effect to trap the atom [107, 108]. The AC Stark effect involves applying a beam, off-resonant to the allowed electronic state transitions, as shown in fig. 1.14. This gives rise to an interaction Hamiltonian, that when diagonalized, results in light dressed states with their energy shifted compared to the bare atomic Hamiltonian. This energy shift is related to the intensity of the laser. Due to the spatial profile of the laser, typically Gaussian, used in the optical tweezer, the atom experiences a corresponding spatially dependent energy shift. Depending on the frequency of the optical tweezer, the current state of the atom and its allowed transitions, the energy shift can either trap the atom at the focal point of the Gaussian beam or push the atom out of the trap entirely.

Qubit scheme

There are typically two qubit schemes in the Rydberg atom array [109]: the Rydberg qubit and hyperfine qubit. The hyperfine qubit scheme is similar to that of the trapped-ions hyperfine qubit scheme. The Rydberg qubit, as shown in fig. 1.15, involves the ground

state and a high lying Rydberg state, i.e. an electronic state with a high principal quantum number. The states in the Rydberg qubit are long-lived and satisfies (1) and (3) of DiVincenzo's criterion.

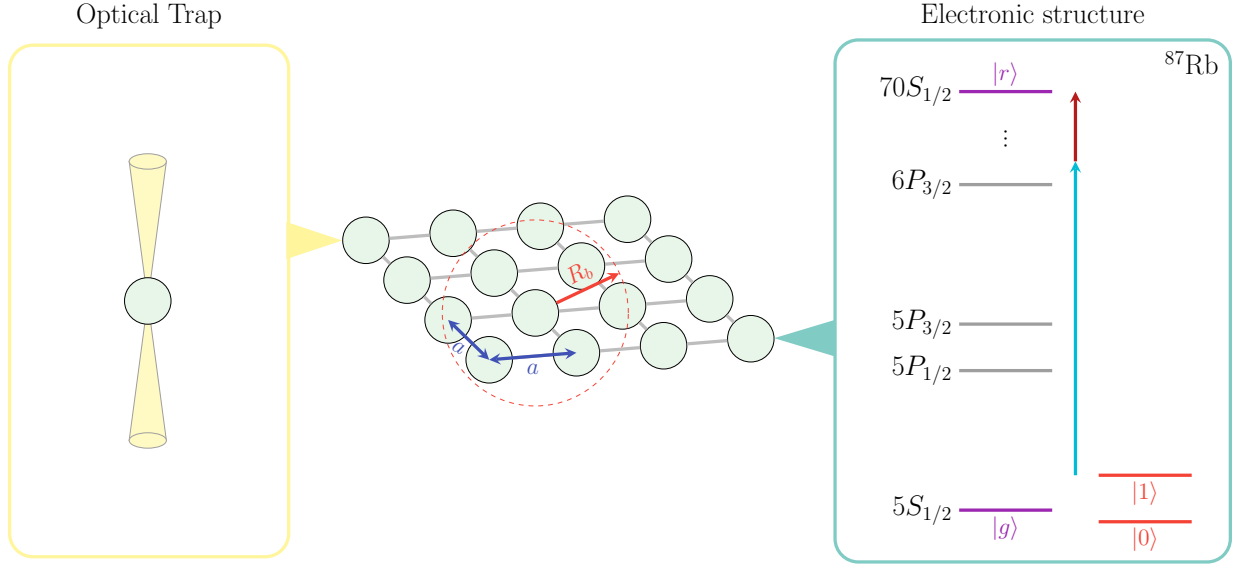


Figure 1.15: **Rydberg atom array quantum processor.** On the left shows a diagram of an optical tweezer trapping an atom. On the right shows the electronic structure of the atom with indicators for the hyperfine and rydberg qubit schemes.

Quantum operations

To prepare the Rydberg atom array, we cool the atoms with doppler cooling (described previously). This removes kinetic energy from the atoms and allows them to remain still at the focal point of the individual optical tweezers. Similar to in trapped-ions, the initial electronic state of the atoms is prepared via optical pumping. Typical measurement of the Rydberg atom array involves driving one of the qubit states to an anti-trapping state resonantly [109]. When in the anti-trapping state, the atom is repelled out of the trap and is lost. A fluorescent transition is then driven and the presence of the atom indicate the results of the measurement. The above procedures satisfy (2) and (5) of DiVincenzo's criterion.

Single qubit gates are performed via the two-photon Raman transition, similar to in

trapped-ions. Whereas, the entangling gate naturally occurs due to the interactions between the electrons when they populate the Rydberg state. The highly excited Rydberg state occupies a relatively large region of space, this forbids other nearby atoms from populating the Rydberg state. This interaction is a dipole-dipole Van der Waals interaction known as the Rydberg blockade [109–113]. These processes satisfy the universal gate set requirement specified by DiVincenzo’s criterion (4).

A typical Hamiltonian of the Rydberg atom array takes the following form:

$$\hat{H} = \sum_{i < j} C_6 V_{ij} \hat{n}_i \hat{n}_j - \delta \sum_i \hat{n}_i - \frac{\Omega}{2} \sum_i \hat{\sigma}_i^x, \quad (1.15)$$

$$C_6 = \Omega \left(\frac{R_b}{a} \right)^6, \quad V_{ij} = \left(\frac{a}{\|\mathbf{r}_i - \mathbf{r}_j\|} \right)^6. \quad (1.16)$$

where $\hat{\sigma}_i^x = |g\rangle_i \langle r|_i + |r\rangle_i \langle g|_i$, the occupation number operator $\hat{n}_i = \frac{1}{2} (\hat{\sigma}_i + \mathbb{1}) = |r\rangle_i \langle r|_i$ and $\hat{\sigma}_i = |r\rangle_i \langle r|_i - |g\rangle_i \langle g|_i$. The experimental settings of a Rydberg atom array are controlled by the detuning of the Raman beams from resonance δ , the effective Rabi frequency Ω of the two-photon Raman transition and the positions of the atoms $\{\mathbf{r}_i\}_i^N$ [11, 92, 93].

1.5 Machine Learning

Machine learning is a framework designed to frame the process of teaching a machine how to process data and make predictions as an optimization problem. Machine learning is heavily inspired by the human brain as can be seen from the naming of the elementary building block of machine learning architectures, the artificial neuron [114].

An artificial neuron represents a parameterized non-linear multi-input scalar function as follow:

$$f_\lambda(\mathbf{x}) = \sigma(W\mathbf{x} + \mathbf{b}). \quad (1.17)$$

Here, σ is a non-linear function and $\lambda = \{W, \mathbf{b}\}$ is the set of parameters comprising of the weights and biases respectively. Common choices for the non-linear activation function are described in table 1.2.

The universal approximation theorem [115, 116] states that these neurons can be combined to approximate any arbitrary function. An architecture for a universal function approximator is the multi-layer perceptron (MLP) [117, 118]. The MLP is a type of feed-forward neural network (FFNN) where neurons are organized into multiple stacked layers and data is propagated sequentially through each layer, as shown in fig. 1.17.

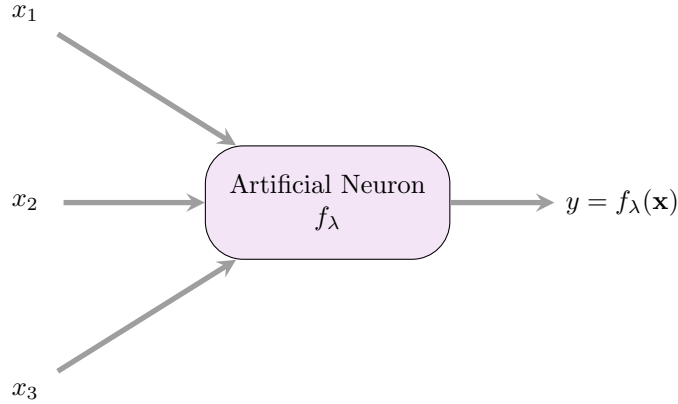


Figure 1.16: **Artificial neuron.** On the left are the inputs to the neuron $\{x_i\}$, the artificial neuron, in general, performs a non-linear function on the inputs and outputs a scalar result.

1.5.1 Backpropagation

A key algorithm used in machine learning is backpropagation [119]. Backpropagation is an algorithm for calculating the derivatives of a composite function with respect to multiple parameters by making use of the chain rule of derivatives:

$$y_n = f_n \circ f_{n-1} \circ \dots \circ f_2 \circ f_1(x), \quad (1.18)$$

$$\frac{\partial y_n}{\partial x} = \frac{\partial f_n}{\partial y_{n-1}} \frac{\partial f_{n-1}}{\partial y_{n-2}} \dots \frac{\partial f_2}{\partial y_1} \frac{\partial f_1}{\partial x}. \quad (1.19)$$

First, we put the composite function into the form of a computational tree with directed edges. Then we calculate the derivatives corresponding to each edge in the tree. The combination of derivatives form the backwards flow through the computational tree. An example application of backpropagation is shown in fig. 1.18.

With backpropagation, we can compute the gradient of the output of the computational tree with respect to any node, particularly the leaf nodes, in the computational tree.

A generalized version of backpropagation to multi-input multi-output functions, $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$, is automatic differentiation (AD) [120]. Automatic differentiation comes in two flavours: forward-mode and reverse-mode.

- In forward-mode AD the iterative unfolding of the derivatives are performed bottom

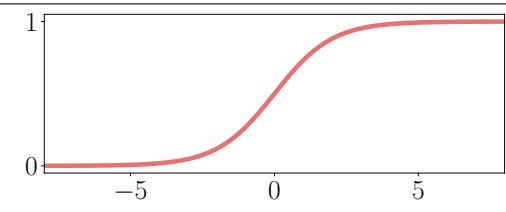
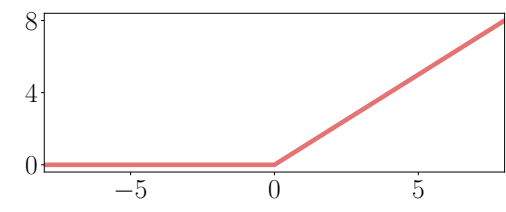
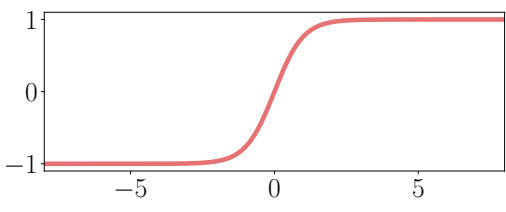
Name	Equation	Plot
Sigmoid	$\frac{1}{1 + e^{-x}}$	
Rectified linear unit (ReLU)	$\max(0, x)$	
Hyperbolic tangent	$\tanh(x)$	

Table 1.2: **Common activation functions used in neural networks.**

to top in the computation tree:

$$\begin{aligned}
\frac{\partial \mathbf{y}_n}{\partial \mathbf{x}} &= \frac{\partial \mathbf{y}_n}{\partial \mathbf{y}_{n-1}} \frac{\partial \mathbf{y}_{n-1}}{\partial \mathbf{x}} \\
&= \frac{\partial \mathbf{y}_n}{\partial \mathbf{y}_{n-1}} \left(\frac{\partial \mathbf{y}_{n-1}}{\partial \mathbf{y}_{n-2}} \frac{\partial \mathbf{y}_{n-2}}{\partial \mathbf{x}} \right) \\
&= \frac{\partial \mathbf{y}_n}{\partial \mathbf{y}_{n-1}} \left(\frac{\partial \mathbf{y}_{n-1}}{\partial \mathbf{y}_{n-2}} \left(\frac{\partial \mathbf{y}_{n-2}}{\partial \mathbf{y}_{n-3}} \frac{\partial \mathbf{y}_{n-3}}{\partial \mathbf{x}} \right) \right) \\
&= \dots
\end{aligned} \tag{1.20}$$

- In reverse-mode AD the iterative unfolding of the derivatives are performed top to

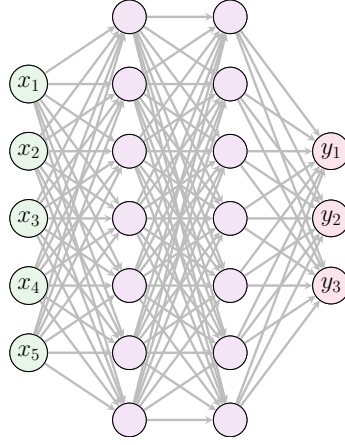


Figure 1.17: **Multi-layer perceptron.** On the left is the input node, followed by layers of hidden neurons and a final layer of output neurons.

bottom in the computation tree:

$$\begin{aligned}
 \frac{\partial \mathbf{y}_n}{\partial \mathbf{x}} &= \frac{\partial \mathbf{y}_n}{\partial \mathbf{y}_1} \frac{\partial \mathbf{y}_1}{\partial \mathbf{x}} \\
 &= \left(\frac{\partial \mathbf{y}_n}{\partial \mathbf{y}_2} \frac{\partial \mathbf{y}_2}{\partial \mathbf{y}_1} \right) \frac{\partial \mathbf{y}_1}{\partial \mathbf{x}} \\
 &= \left(\left(\frac{\partial \mathbf{y}_n}{\partial \mathbf{y}_3} \frac{\partial \mathbf{y}_3}{\partial \mathbf{y}_2} \right) \frac{\partial \mathbf{y}_2}{\partial \mathbf{y}_1} \right) \frac{\partial \mathbf{y}_1}{\partial \mathbf{x}} \\
 &= \dots
 \end{aligned} \tag{1.21}$$

The decision of which flavour of AD should be used depends on the dimensions of the input and output of the function [120, 121].

1.5.2 Gradient-based optimization

To perform training on the machine learning models the model parameters λ are adjusted in accordance to the optimization of a loss function $\mathcal{L} : \mathbb{R}^N \rightarrow \mathbb{R}$.

$$\operatorname{argmin}_{\lambda} \mathcal{L}(M_{\lambda}, \dots), \tag{1.22}$$

where, M_{λ} represents the model.

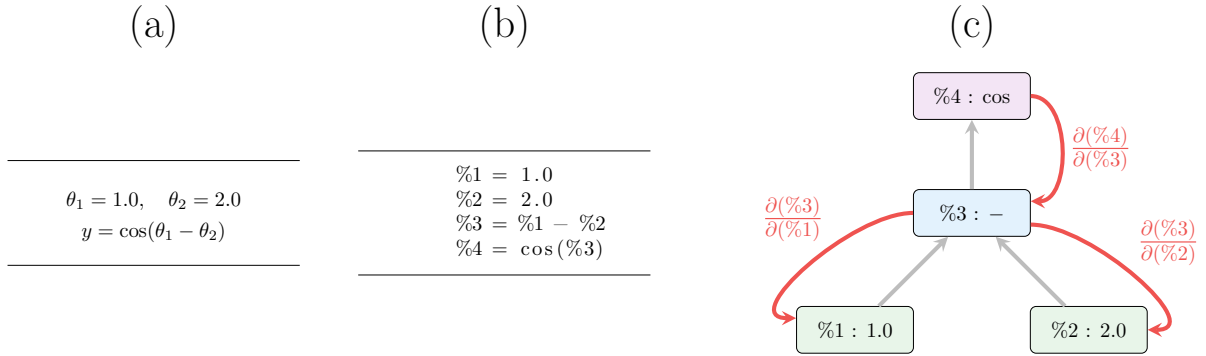


Figure 1.18: **Backpropagation.** (a) Mathematical representation, (b) breakdown of computational steps and (c) computation tree.

The strategy to optimize the parameters involve using the gradient of the loss function with respect to the model parameters:

$$\lambda' = \lambda - \eta \nabla_{\lambda} \mathcal{L}, \quad (1.23)$$

where η is the learning rate. The gradient can be calculated efficiently with the backpropagation introduced in fig. 1.18. This is the gradient descent algorithm [122].

There are a multitude of variation of the gradient descent algorithms that seek to address its shortcomings by introducing momentum [123, 124] and other techniques, they include RMSprop [125], AdaGrad [126], Adam [127], AdaDelta [128], etc.

1.6 Generative models

Generative models is a sub-category of machine learning, where the architecture is designed to perform generative tasks. Generative models parameterize a joint probability distribution of multiple parameters and are equipped with an algorithm for sampling the parameters from the parameterized distribution. In recent years, generative models have demonstrated their ability to capture complex probability distributions. Generative models have been used for many task, such as natural language processing, image generation, etc.

Training these generative models involve the optimization of the parameterized probability distribution to match that of the target distribution. To characterize the distance between

the distributions, we use a statistical distance such as the Kullback-Leibler divergence (KLDiv) [129]:

$$\text{KLDiv}(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \left(\frac{p(\mathbf{x})}{q(\mathbf{x})} \right). \quad (1.24)$$

1.6.1 Energy-based models (EBMs)

Energy-based models (EBMs) are a family of physics-inspired machine learning models. These models may resemble many-body physics models like those described in section 1.1. In these models, an energy function E_{λ} , is associated with the configuration of the network with parameters λ . This energy function is applied to the inputs $\mathbf{x}$ to obtain the energy of this state.

Hopfield network

The Hopfield network [130] was a pioneer in the class of energy-based models. The Hopfield model is an adaptation of the well studied Ising model to be a computational tool. In particular, the Hopfield model was designed to store a content-addressed memory, also known as an associative memory [130].

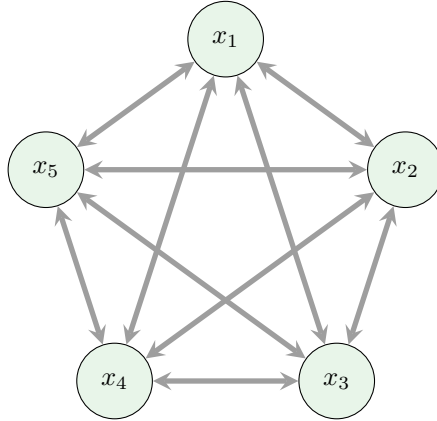


Figure 1.19: **Hopfield model.** The model consist of full interactions between the neurons, resembling an infinite dimensional Ising model, the strength of each bond is parameterized.

The Ising model is a many-body system comprising of spins x_i that is able to take on one

of two states. The energy of the Ising model takes on the following form:

$$E_W(\mathbf{x}) = - \sum_{i < j} W_{ij} x_i x_j, \quad x_i \in \{\pm 1\}, \quad (1.25)$$

where x_i is the i -th spin and W_{ij} indicates the coupling between i -th and j -th spin. For the Hopfield model, self-connections are prohibited, i.e.

$$W_{ii} = 0 \quad \forall i. \quad (1.26)$$

The associative memory is encoded with the many-body system by adjusting the parameters W such that the memory state is the minimum energy state of the energy. The memory can then be retrieved from any other state via the minimization of the energy.

The first algorithm used to train the Hopfield network is Hebbian learning. This originates from the observation that the energy eq. (1.25) takes on a quadratic form. As such, for a single memory state, we set the parameters W to be the projection matrix after applying eq. (1.26). For p memory states this is generalized by summing the respective projection matrices, i.e.

$$W_{ij} = \frac{1}{p} \sum_{\mathbf{x}} x_i x_j \quad \forall i \neq j. \quad (1.27)$$

Hebbian learning has a severe drawback in the case of multiple memory states as they lead to local minima states, in the energy, that corresponds to states not within the desired memory states. Other training scheme have been developed to circumvent this issue [131]. However, Hopfield models have a critical drawback as they are provably only able to encode a linear amount of patterns for a given system size [130].

Boltzmann machine

Boltzmann machine [132] are a generalization of the Hopfield model that introduces hidden units alongside the usual visible units. The energy of the Boltzmann machine takes on the following form:

$$E_\lambda(\mathbf{x}) = \sum_{i < j} W_{ij} x_i x_j + \sum_i b_i x_i, \quad x_i \in \{\pm 1\}, \quad (1.28)$$

$$\lambda = (W, \mathbf{b}), \quad \mathbf{x} = (\mathbf{v}, \mathbf{h}), \quad (1.29)$$

where $\mathbf{v}, \mathbf{h}$ are the visible and hidden units respectively.

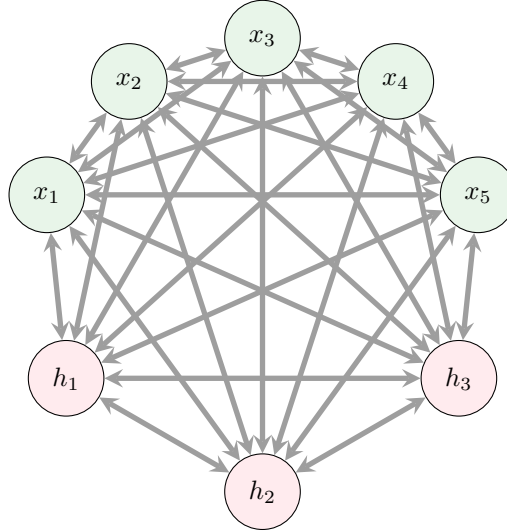


Figure 1.20: **Boltzmann machine.** The introduction of hidden units h_i improves the capacity of the Boltzmann model compared to the hopfield network. The system is fully-connected between all visible and hidden units.

The Boltzmann machine draws further connection to many-body physics by integrating the Boltzmann distribution into the architecture.

$$p_{\lambda}(\mathbf{x}) = \frac{e^{-E_{\lambda}(\mathbf{x})}}{\mathcal{Z}}, \quad \mathcal{Z} = \sum_{\sigma} e^{-E_{\lambda}(\sigma)}. \quad (1.30)$$

The model considers the visible and hidden units to be drawn from the Boltzmann distribution. The procedure for sampling the visible and hidden units from the Boltzmann machine takes the form of the Markov chain Monte-Carlo algorithm described in section 1.3.1.

The Boltzmann machine is trained to replicate the underlying distribution of some dataset. This is done by maximizing the likelihood of observing the dataset. The Boltzmann machine allows access to the unnormalized probability distribution through the energy function eq. (1.29), however normalizing this probability distribution is generally intractable due to the requirement of calculating the partition function $\mathcal{Z}$. Without access to the normalized probability distribution, training the Boltzmann machine via maximizing the likelihood is inefficient.

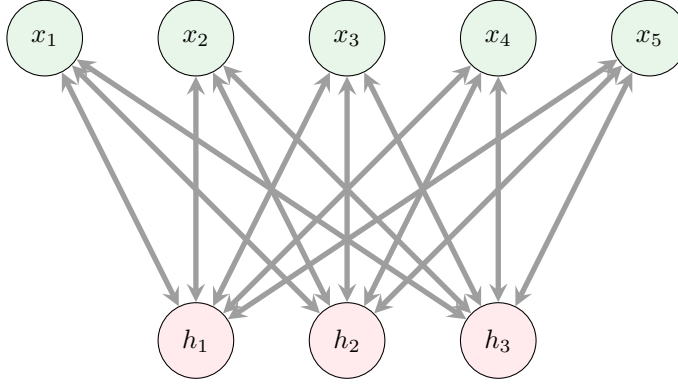


Figure 1.21: **Restricted Boltzmann machine (RBM)**. The bonds in the RBM are restricted to form a bipartite graph, where the system is split between the visible and hidden units.

Restricted Boltzmann machine (RBM)

The restricted Boltzmann machine (RBM) is a modification of the Boltzmann machine such that the connections form a bipartite graph between the visible and hidden units of the system. The energy for the RBM takes the same form as that of the Boltzmann machine eq. (1.29).

Similar to the Boltzmann machine training the RBM via the maximization of the probability is inefficient due to the intractability of the partition function $\mathcal{Z}$. Therefore, the RBM is instead trained via a different procedure contrastive divergence [133].

RBMs also allow a more efficient sampling procedure based on Gibbs sampling [134], k -block Gibbs sampling. This procedure is more efficient compared to the sampling procedure for the Boltzmann machine.

Due to the fact that Monte-Carlo methods are used to sample the RBM, there is a natural autocorrelation time between samples for them to be fully uncorrelated. This affects the efficiency of sampling for these models.

1.6.2 Flow-based models

In the above energy-based models, the distribution captured by the system is limited. In this section, we describe a way to parameterize a more general probability distribution

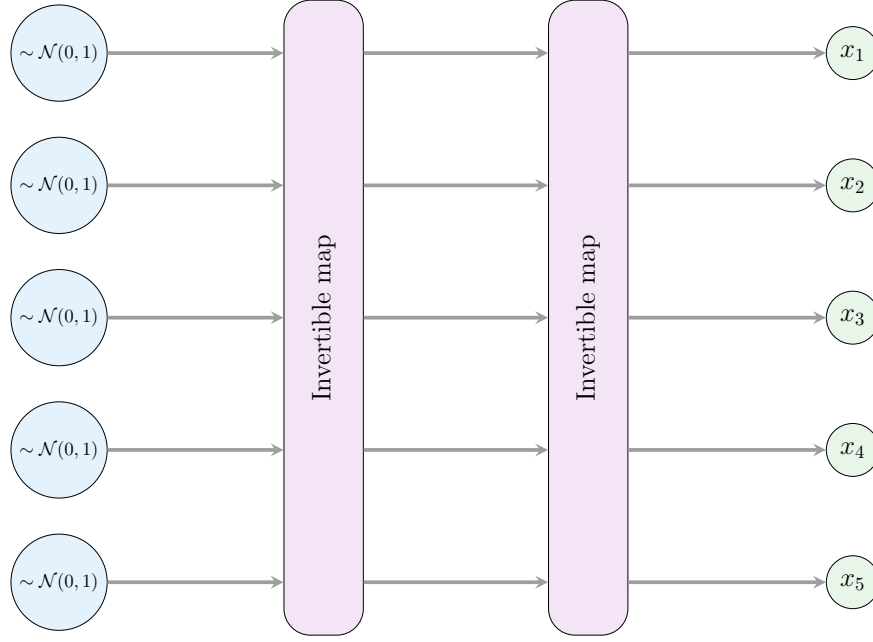


Figure 1.22: **Flow-based model.** The left blue units represent the samples taken from an initial distribution, a typical choice being the standard normal distribution. The right green units represent samples drawn from the flow model.

using bijective, i.e. invertible, transformations, this architecture is known as flow-based models.

Flow-based models comprise of an initial distribution and a set of parameterized bijective transformations. A typical choice for the initial distribution is Gaussian distribution and its generalizations, this forms the subcategory of models known as normalizing flows [135–137]. Figure 1.22 shows the architecture of a flow based model.

Sampling from the flow-based models involves first sampling from the initial distribution. Then, the sample is transformed via the parameterized bijective functions. The output of the process forms a sample of the flow-based model.

$$\mathbf{x} = f_l \circ f_2 \circ \dots \circ f_{l-1} \circ f_l(\mathbf{z}_0), \quad (1.31)$$

where, f_i is the i -th bijective transform layer and z_0 the sample from the initial distribution and x the sample from the model. Typical parameterized bijective functions include different types of splines [138–140] and affine transformations [141].

1.6.3 Autoregressive models

In the earlier introduced generative models, the joint probability distribution is parameterized in its entirety. In this section, we introduce an alternative method for the parameterization of a joint probability distribution via the autoregressive decomposition of probability distributions.

In statistics, a well-known theorem is the chain rule for probabilities:

$$\begin{aligned} p(\mathbf{x}) &= p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)\dots p(x_N|x_1, x_2, \dots, x_{N-1}) \\ &= \prod_{n=1}^N p(x_n|\mathbf{x}_{<n}) \end{aligned} \tag{1.32}$$

This theorem states that any joint probability distribution can be decomposed into a set of conditional probability distributions.

Via the autoregressive decomposition of joint probability distribution, independent and identically distributed (IID) random variables can be easily drawn via the continued sampling of the ordered set of conditional probability distributions.

The set of conditional probabilities naturally form a sequence where the order is governed by causality. As such, the set of models most appropriate for parametrizing the autoregressive probability distribution are sequential models, such as recurrent neural networks (RNNs), transformers, etc.

The process of outfitting sequential models with the autoregressive property involves ensuring the sequence obeys causality, i.e. the current outcome only depends on the outcomes preceding it. The output of the sequential model also has to be converted to a discrete probability distribution by applying a generator layer. The generator typically takes the form of a dense layer with a softmax function applied to the output.

$$\text{Softmax}(\mathbf{x})_i = \frac{e^{x_i}}{\sum_j e^{x_j}}. \tag{1.33}$$

Recurrent Neural Network (RNN)

Recurrent Neural Networks (RNNs) [142] are a sequential model with recurrent connection. These recurrent connection connects outputs back into the input of a computation cell. This recurrent information is the hidden state of the RNN and stores context for the

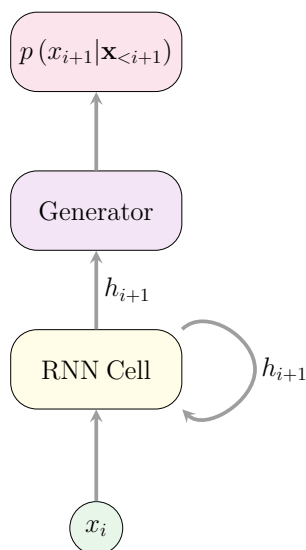


Figure 1.23: **Recurrent neural network.** The recurrent cell takes as input the current hidden state and an element of the input sequence. The output of the recurrent cell is used both as the new hidden state and also as an element of the output sequence of the RNN. As the recurrent cell is called multiple times, the input sequence is consumed and processed into an output sequence. The output sequence is converted to a sequence of conditional probabilities by a generator.

previous runs of the computation cell, each of these runs update the state. The RNN is typically initialized with a zero vector as the hidden state.

To autoregressively encode a joint probability distribution, we start the RNN with a fixed initial token x_0 , instead of the first element of the input sequence x_1 , this allows the first element of the output sequence to be the probability distribution governing x_1 . Next, the first element x_1 is fed, alongside the updated hidden state, to the RNN cell to produce the next element of the output sequence, that only depends on x_1 . This process is repeated with the following input token, as shown in fig. 1.23. The output only depends on the inputs preceding it satisfying the autoregressive property. The output sequence of the RNN is then fed into the previously mentioned generator layer and mapped to conditional probability distributions.

An initial problem faced by RNNs, is the problem of exponential growth and decay of gradients during the training process [143, 144]. Due to the iterative nature of the RNN, the gradient of the parameters affecting the early units of the RNN are suppressed or exploded leading to vanishing or unstable training respectively. To remedy the problem,

gated recurrent unit (GRU) cell [145], long short-term memory (LSTM) cell [146], etc., were developed.

The representation power of the RNN is governed by the size of the hidden state. As the hidden state is constantly updated and maintains the same size throughout the unrolling of the model, the past context is necessarily being compressed. This leads to an exponential decay of correlations between units far apart in the sequence length [144, 147]. One way to increase the power of RNNs is to consider passing the hidden state at multiple points instead of only passing the previous hidden state [148]. However, necessarily, the temporal connections must obey causality to preserve the autoregressive property of the model.

The computational complexity of the RNN depends linearly on the sequence length and quadratically on the size of the hidden state. This makes the RNNs a highly efficient architecture for predicting long sequences.

Transformer

Transformers [149] are a sequential model that takes advantage of an attention mechanism. The attention mechanism relates the importance between pairs of points in the sequence. These attention scores are computed with the scaled dot product attention:

$$A = \frac{QK^T}{\sqrt{d_k}}V. \quad (1.34)$$

Given that the attention connect all possible pairs in the sequence, transformers are naturally long-range. This full connection in the transformer results in the computational complexity scaling quadratically with the sequence length [149, 150].

Instead of performing attention on the entirety of the embedding space, they can be parallelized by splitting the embedding space and applying the scaled dot product attention on each subspace individually, this results in the multi-head attention (MHA). The computational complexity of the multi-head attention mechanism is linear in the size of the embedding space.

As a consequence of the full connections present, the transformer is not autoregressive. The autoregressive property can be restored for the transformer by applying a causal mask to the attention disallowing causally forbidden connections.

Similar to the RNN, the transformer’s input sequence is padded with an initial token x_0 , when autoregressively encoding a joint probability distribution. Since the attention mechanism does not capture any information of the position of tokens in the input sequence,

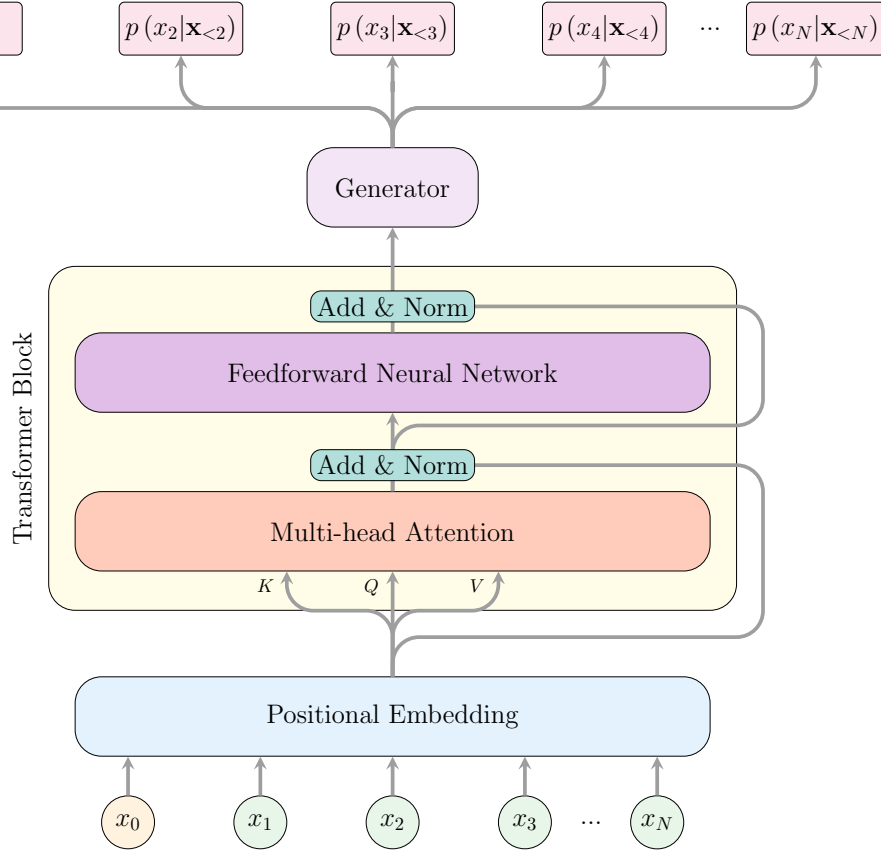


Figure 1.24: **Transformer.** x_0 is the initial token required to initiate the Transformer, typically taken to be the zero vector.

the positional embedding is used to incorporate positional information. The embedded sequence is then processed by the MHA and a FFNN. The output sequence of the transformer is then fed into a generator, mapping it to conditional probability distributions.

1.7 Thesis Outline

In this thesis, we

- In chapter 2: We compare and evaluate the efficiency of classical shadow and neural quantum state (NQS) methods of estimating observables from data. Furthermore,

we introduce the conditional local density RNN (CLDRNN) that seeks to remedy the exponential cost associated with the rotations of NQS for arbitrary Pauli basis measurements. NQSs demonstrate a significant difference in the sampling complexity of estimating observables as compared to the other methods, especially with high-weight Pauli observables. However, this accompanies an increased computational cost for general quantum states. The CLDRNN underperforms due to the biases in the estimations generated by neural networks. Overall, we demonstrate a tradeoff in sampling complexity and computational complexity between methods that incorporate the constraints of quantum states, NQSs, and methods that violate the physicality condition of quantum states, classical shadows and CLDRNN.

- In chapter 3: Autoregressive models have a degree of freedom in the autoregressive decomposition of the joint probability. The 1D sequence corresponding with the autoregressive path affects the learning characteristics of the model, independent of architecture, RNN or transformer. Specifically, the correlations in the model are biased and this bias is related to the properties of the path chosen. Choosing the appropriate path could improve the efficiency of the training of the autoregressive model. For the 2D Ising model with periodic boundary conditions (PBCs), we find that the zigzag path converges the fastest despite its anisotropy that is in direct contrast to the symmetries inherent to the model.
- In chapter 4: NQSs have been trained to reproduce single states, however, with an encoder-decoder structure, we are able to condition these NQSs on system variables, enabling the model to parameterize a family of states. A significant area where this is desirable is with the ground states of parameterized Hamiltonians, such as the Hamiltonian for the Rydberg atom array quantum processor. The model shows signatures of generalization for particular system parameters. However, significant progress is required before these models are able to provide accurate estimates of the properties for a wide range of system variables.
- In chapter 5: Traditionally, autoregressive models such as the transformer has been designed for discrete local DoFs, however, we propose a simple modification, with normalizing flows, that allow these models to parameterize joint probability distribution of continuous variables. We demonstrate this architecture on the classical XY model with the well-known Kosterlitz-Thouless transition. The model is able to parameterize the equilibrium configuration to a high degree across the phase transition. However, once again, we demonstrate the incapability of the transformer at generalizing to larger systems.

- In chapter 6: We develop an architecture, called the fractal transformer, introducing a hierarchical structure to the transformer inspired by Kadanoff RG and the efficiency of the patched transformer. Our results demonstrate that the fractal transformer is able to better generalize to larger system sizes for systems with local order parameters, such as the Ising model and the XY model. The model is conducive to the transfer learning process, where models are initially trained on smaller system sizes and fine-tuned for larger system sizes. However, the model fails to perform for systems without local order parameters like the Ising lattice gauge theory.

Chapter 2

Estimating physical observables from data

Quantum processors, as described in section 1.4, have been developed under the speculation that they lead to computational advantages in a variety of problems. One such area where quantum processors distinguish themselves is in the quantum simulation of naturally quantum systems such as materials [88], molecules [151], etc. In quantum simulations experiments, the quantum simulator serves as a surrogate for the quantum system of interest. In order to learn about the quantum system of interest, we extract information from the surrogate quantum simulator by measuring the system. However, due to the probabilistic nature of quantum systems, singular measurements of the quantum processor does not result in the full information about the current state of the system. Therefore, we must look to extract the information of interest from the quantum system via repeated measurements of the quantum system, in general these measurements are done in a multitude of bases.

From these measurements, information of the system can be extracted with the following procedures:

- Classical shadows [152]
- Quantum state tomography (QST)

Performing QST for the full state is known to be intractable due to the exponential growth of the number of complex parameters required to express a quantum state as the system grows in size. To address this issue, methods for performing QST with a compressed representation of the state were developed. One such approach to the problem is with

a tensor networks such as matrix product states (MPSs) [153, 154], projected entangled pair states (PEPSs) [155], tree tensor network (TTN) states [156], etc. These tensor product states can be optimized to find the ground state of a system via density matrix renormalization group (DMRG) algorithm, a variational method for searching the space of a particular tensor product ansatz [157–160]. These methods are hugely successful and still remain state of the art for 1-dimensional (1D) systems. However, the space of states efficiently expressible by these tensor network ansatz are provably incomplete. In particular, given a fixed bond dimension, the entanglement of these states follow an area-law [155, 161].

Taking inspiration from machine learning algorithms’ capabilities in compression (autoencoder models) and the non-linearity in such models, neural quantum states (NQSs) were developed with the prospect of being able to efficiently represent a wider variety of states including certain volume law entangled states [162, 163]. Due to the probabilistic nature of quantum states, a natural choice for NQSs are generative models, such as those outlined in section 1.6.

In general, classical shadow outperform QST methods in computational complexity due to their simplicity. However, it has been shown that this comes with a tradeoff in sampling complexity [164], i.e. classical shadows require more data for a desired accuracy of the observable.

In this thesis, we are interested in two platforms for quantum simulation trapped-ions and rydberg atom arrays. Trapped-ions and rydberg atom arrays have a repetition rate of approximately 10^2 Hz and 1 Hz respectively, as such data generation from these machines is expensive and limited. This repetition rate is governed by the protocols in such systems, preparation of the initial state and measurements are of order 10^{-3} s to 10^{-2} s and gate times are of order 10^{-4} s to 10^{-5} s. For the Rydberg atom array, the measurements results in the lost of atoms leading to the slower repetition rate. Further optimizations of the experiment could push the repetition rate by another order of magnitude. Therefore, for these machines, the ability to extract the maximal amount of information from the limited data is crucial.

In this chapter, we seek to compare the sampling complexity of classical shadows and complex RNN wavefunction for a typical trapped-ion quantum simulation experiment. We propose the conditional local density RNN (CLDRNN) in an attempt to address the issue with exponential computational complexity of training the complex RNN wavefunction ansatz. Furthermore, we demonstrate the mechanism that leads classical shadows to have a higher sampling complexity for certain operators.

2.1 Classical shadows

Given that physical observables are Hermitian and the Pauli operators form a basis for Hermitian operators, all observables can be decomposed into a linear combination of Pauli operators and written as:

$$\hat{O} = \sum_Q O_Q \hat{Q}, \quad (2.1)$$

$$\hat{Q} = \hat{Q}_1 \hat{Q}_2 \dots \hat{Q}_N, \quad \hat{Q}_i \in \{\hat{1}, \hat{X}, \hat{Y}, \hat{Z}\}. \quad (2.2)$$

As such, if we obtain an estimate for the expectation of all Pauli operators, we are able to construct an estimate for the expectation of arbitrary physical observables.

$$\langle \hat{O} \rangle = \sum_Q O_Q \langle \hat{Q} \rangle. \quad (2.3)$$

Classical shadow algorithm [152, 165] makes use of random uniform Pauli measurements to estimate the expectation of all Pauli operators.

A Pauli measurement is a measurement of the system in a pauli basis B .

$$B = B_1 B_2 \dots B_N, \quad B_i \in \{X, Y, Z\}. \quad (2.4)$$

To perform this measurement, typically, appropriate local rotations are applied to each qubit to change basis to the desired Pauli basis B_i . The Pauli basis rotations take the following form:

$$\begin{aligned} \hat{U}_X &= \hat{H}, \\ \hat{U}_Y &= \hat{H} \hat{S}^\dagger, \\ \hat{U}_Z &= \hat{1}. \end{aligned} \quad (2.5)$$

Here, $\hat{H}$ and $\hat{S}$ are the Hadamard and phase gate respectively. After applying the appropriate gates, the qubits then undergo a projective measurement in the computational basis, typically taken to be $B = ZZ \dots Z$. The outcome of the Pauli basis measurement takes the following form

$$\mathbf{M} = \{\sigma, B\}, \quad (2.6)$$

$$\sigma = \sigma_1 \sigma_2 \dots \sigma_N, \quad \sigma_i \in \{0, 1\}. \quad (2.7)$$

where σ is the measurement outcome.

To outline the classical shadows algorithm, we define the following helpful terms:

$$\text{Supp}(\hat{Q}) = \{i | \hat{Q}_i \neq \hat{1}\}, \quad (2.8)$$

$$\text{Weight}(\hat{Q}) = |\text{Supp}(\hat{Q})|, \quad (2.9)$$

$$f_i(B, \hat{Q}) = \begin{cases} 1 & \text{if } \hat{Q}_i = \hat{1} \\ 3 & \text{if } B_i = \hat{Q}_i \\ 0 & \text{if } B_i \neq \hat{Q}_i \end{cases} \quad (2.10)$$

$$\mu_i(\sigma) = \begin{cases} -1 & \text{if } \sigma_i = |0\rangle \\ +1 & \text{if } \sigma_i = |1\rangle \end{cases} \quad (2.11)$$

The classical shadows algorithm to estimate a Pauli observable $\hat{Q}$ from a set of random uniform Pauli measurements $\mathcal{D} = \{\mathbf{M}_d\}_{d=1}^D$ is as follows:

$$\langle \hat{Q} \rangle = \frac{1}{D} \sum_{\mathbf{M} \in \mathcal{D}} \prod_i f_i(B, \hat{Q}) \prod_{j \in \text{Supp}(\hat{Q})} \mu_j(\sigma). \quad (2.12)$$

A caveat with the classical shadows algorithm is that the estimates generated are not guaranteed to be compatible with a physical state [152].

2.2 Neural quantum states (NQSs)

Neural quantum states (NQSs) can be parameterized by energy-based models [166–173], sequential models [174, 175] and autoregressive models [164, 176–188] described in sections 1.6.1 and 1.6.3 respectively. In this chapter, we adopt the architecture described in the paper *Recurrent neural network wave function* [176] where the NQS is parameterized by an autoregressive RNN model (shown in fig. 2.1).

The following equations are used to obtain the quantum state parameterized by the complex RNN wavefunction:

$$|\Psi_{\text{NN}}\rangle = \sum_{\sigma} \Psi_{\text{NN}}(\sigma) |\sigma\rangle, \quad (2.13)$$

$$\Psi_{\text{NN}}(\sigma) = e^{i\phi_{\text{NN}}(\sigma)} \sqrt{p_{\text{NN}}(\sigma)}, \quad (2.14)$$

$$p_{\text{NN}}(\sigma) = \prod_i p(\sigma_i | \sigma_{<i}), \quad \phi_{\text{NN}}(\sigma) = \sum_i \phi_i(\sigma_i | \sigma_{<i}) \quad (2.15)$$

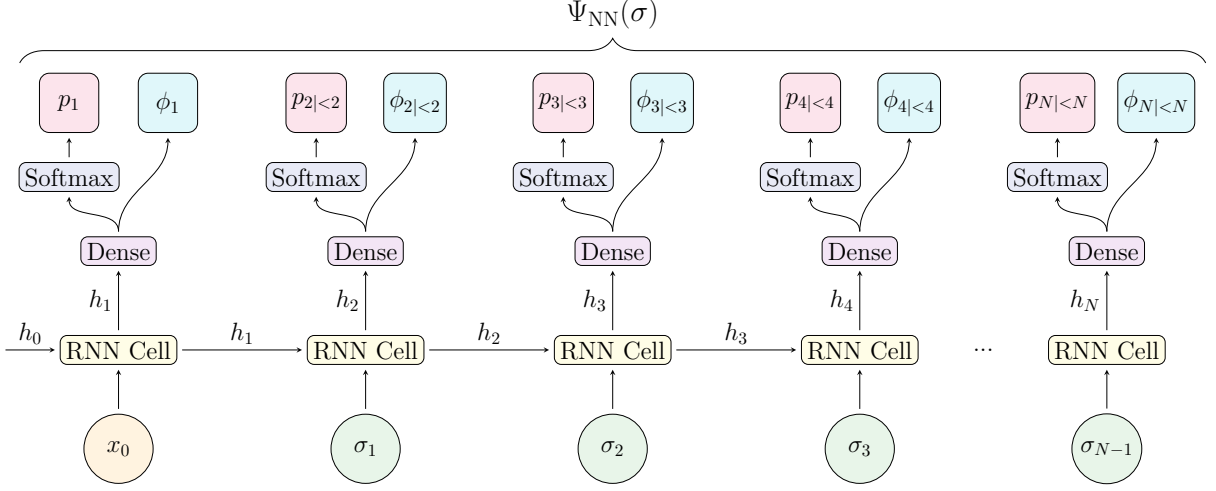


Figure 2.1: **Complex RNN wavefunction.** h_0 and x_0 are the initial hidden state and initial token respectively required to initiate the RNN, they are typically taken to be the zero vector.

As can be seen from eq. (2.15), obtaining the probabilities in the computational basis is simple. However, obtaining the probabilities of the state in an arbitrary Pauli basis is inefficient and takes the following form:

$$p(\mathbf{M}) = \langle \sigma | \hat{U}_B^\dagger | \Psi_{NN} \rangle, \quad (2.16)$$

where, $\mathbf{M}$ denotes a measurement outcome σ in the Pauli basis B (eq. (2.7)). In general, $\hat{U}_B$ is a dense operator, as such all coefficients of the state needs to be evaluated to perform calculation. This results in an exponential scaling with the number of qubit when evaluating the probability of a particular outcome from a Pauli measurement.

Computational basis samples can be obtained from the complex RNN wavefunction with the following method:

1. Feed an initial token x_0 to the complex RNN wavefunction, the outputs include the probabilities $p(\sigma_1)$ and corresponding local phases $\phi(\sigma_1)$ for both states of the first qubit.
2. Sample from the probabilities $p(\sigma_1)$ of the first qubit, the outcome is the measurement of the first qubit.

3. Repeat step 1 replacing the initial token with the outcome of the measurement of the first qubit σ_1 , the outputs include the conditional probabilities $p(\sigma_2|\sigma_1)$ and corresponding local phases for both states of the second qubit $\phi(\sigma_2|\sigma_1)$.
4. Step 2 can be repeated for the second qubit to obtain the outcome of the measurement of the second qubit σ_2 .
5. Continue to repeat steps 3 and 4 with the next qubit in the sequence until the last qubit is measured.

Sampling from an arbitrary Pauli basis requires the probabilities of every state of the Pauli basis to be evaluated in accordance to eq. (2.16). As aforementioned, obtaining the probabilities in an arbitrary Pauli basis requires the state to be fully evaluated. Therefore, performing Pauli measurement of the complex RNN wavefunction is intractable.

We use the local estimators to estimate the expectation value of observables $\hat{O}$ from the complex RNN wavefunction [169, 176, 189, 190].

$$\begin{aligned}
\langle \hat{O} \rangle_{\text{NN}} &= \langle \Psi_{\text{NN}} | \hat{O} | \Psi_{\text{NN}} \rangle, \\
&= \sum_{\sigma, \sigma'} \langle \Psi_{\text{NN}} | \sigma \rangle \langle \sigma | \hat{O} | \sigma' \rangle \langle \sigma' | \Psi_{\text{NN}} \rangle, \\
&= \sum_{\sigma, \sigma'} |\langle \Psi_{\text{NN}} | \sigma \rangle|^2 \langle \sigma | \hat{O} | \sigma' \rangle \frac{\langle \sigma' | \Psi_{\text{NN}} \rangle}{\langle \sigma | \Psi_{\text{NN}} \rangle}, \\
&= \sum_{\sigma \sim p_{\text{NN}}} \sum_{\sigma'} O_{\sigma\sigma'} \frac{\Psi_{\text{NN}}(\sigma')}{\Psi_{\text{NN}}(\sigma)}. \\
&= \sum_{\sigma \sim p_{\text{NN}}} O_{\text{loc}}(\sigma)
\end{aligned} \tag{2.17}$$

Here, $O_{\text{loc}} = \sum_{\sigma'} O_{\sigma\sigma'} \Psi_{\text{NN}}(\sigma') / \Psi_{\text{NN}}(\sigma)$ is a local estimator for the observable $\hat{O}$. For a general observable, the local estimator cannot be efficiently evaluated due to the as $O_{\sigma\sigma'}$ is dense. However, if the observable $\hat{O}$ has sparse off-diagonal elements in the computational basis, the local estimator and therefore the expectation is tractable and can be efficiently evaluated.

2.3 Conditional local density RNN (CLDRNN)

To try and overcome the intractable rotations mentioned in section 2.2, we instead look to consider each measurement of a qubit in the system as a measurement of a local density operator, as local density operators can be rotated efficiently.

2.3.1 Local density operators

To determine the local density matrix of a quantum state, we take the partial trace.

Local density operator:

$$\mathbf{A} = A_1 A_2 \dots A_N, \quad (2.18)$$

$$\hat{\rho}_{A_i} = \text{tr}_{\mathbf{A}_{\neq i}}(\hat{\rho}). \quad (2.19)$$

Here, $\mathbf{A}$ represents the full system.

For a single two-level system, an arbitrary density matrix takes the following form:

$$\hat{\rho} = \frac{1}{2} (\hat{\mathbb{1}} + \mathbf{n} \cdot \hat{\boldsymbol{\sigma}}). \quad (2.20)$$

For a product state, it is trivial to prove that with the set local density operators, one is able to scalable obtain the Pauli measurements from the state via local rotations. However, for a more general state, rotation of the set of local density matrices does not yield the same state as the rotation of the global density matrix. As such, one is unable to obtain Pauli measurements of the state without performing the rotation on the entire state. Therefore, the use of local density matrices does not address the problem regarding the requirement of intractable rotations during Pauli measurements.

2.3.2 Conditional local density operators

Instead of obtaining the local density matrices of the state before the Pauli measurement, let us consider the circuit for performing a Pauli measurement (fig. 2.2(a)). A Pauli projective measurement takes the following form:

$$\Pi_{\mathbf{M}} = \hat{U}_B |\sigma\rangle\langle\sigma| \hat{U}_B^\dagger, \quad (2.21)$$

$$p(\mathbf{M}) = \text{tr}(\hat{\Pi}_{\mathbf{M}} \hat{\rho}), \quad (2.22)$$

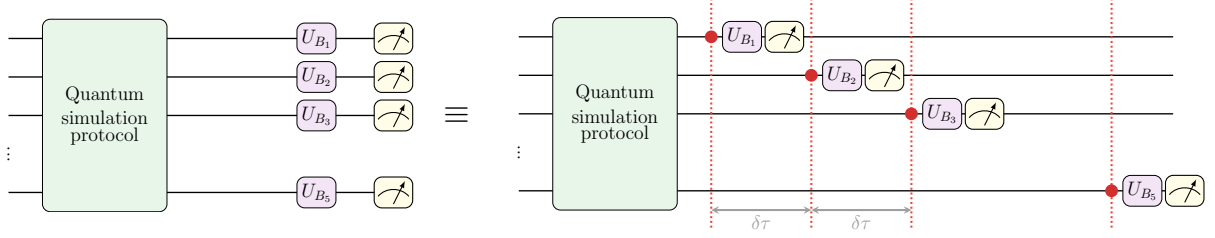


Figure 2.2: **Circuit diagram for conditional local density.** Red dots signify where the conditional local density operators are parameterized by the neural network.

where, $\mathbf{M}$ denotes a measurement outcome σ in the Pauli basis B (eq. (2.7)).

Instead of synchronous Pauli measurement involving parallel applications of the local rotations and measurements, considering staggering them as shown in fig. 2.2(b). In the regime where there are no errors during the time of measurement or the duration of the pauli measurement of a single qubit approaches zero, we see that the synchronous Pauli measurement and the sequential Pauli measurement leads to the same results, samples are drawn from equivalent distributions.

Consider the measurement of the first qubit during the sequential Pauli measurement. This corresponds to evaluating the local density for the first qubit and performing a local rotation and measurement accordingly. The outcome of this measurement naturally affects the remaining state of the system as it projects the first qubit into a particular state. Specifically, the remaining state of the system takes the following form:

$$\begin{aligned}\hat{\rho}_{\mathbf{A}|\mathbf{M}_{<i}} &= \frac{\hat{\Pi}_{\mathbf{M}_{<i}} \hat{\rho} \hat{\Pi}_{\mathbf{M}_{<i}}}{\text{tr} \left(\hat{\Pi}_{\mathbf{M}_{<i}} \hat{\rho} \mathbf{A}_{<i} \right)} \\ &= \mathbf{M}_{<i} \otimes \hat{\rho}_{\mathbf{A}_{\geq i}|\mathbf{M}_{<i}}.\end{aligned}\tag{2.23}$$

For the measurement of the second qubit, we can calculate the conditional local density of the second state dependent on the outcome of the measurement on the first qubit. The conditional local density for the second qubit is as follows:

$$\hat{\rho}_{A_i|\mathbf{M}_{<i}} = \text{tr}_{\mathbf{A}_{>i}} \left(\hat{\rho}_{\mathbf{A}_{\geq i}|\mathbf{M}_{<i}} \right).\tag{2.24}$$

As such, we are able to perform a measurement on the second qubit. Similar to before, the remaining state of the system is now conditioned on the first and second qubits' mea-

surement outcome. This process is continued until all qubits are exhausted and we obtain a Pauli measurement of the entire system.

As previously mentioned, the outcomes are identical to the synchronous measurement, however at each qubit's measurement, we get a conditional local density matrix that we can perform local rotations on and measure. This formulation of the state supports tractable Pauli measurements of the system.

2.3.3 Architecture

To utilize the conditional local density formulation in a neural network, we output the conditional local density matrix after each recurrent step instead of the conditional probabilities and the local phase (as shown in fig. 2.3). We call this network the conditional local density recurrent neural network (CLDRNN).

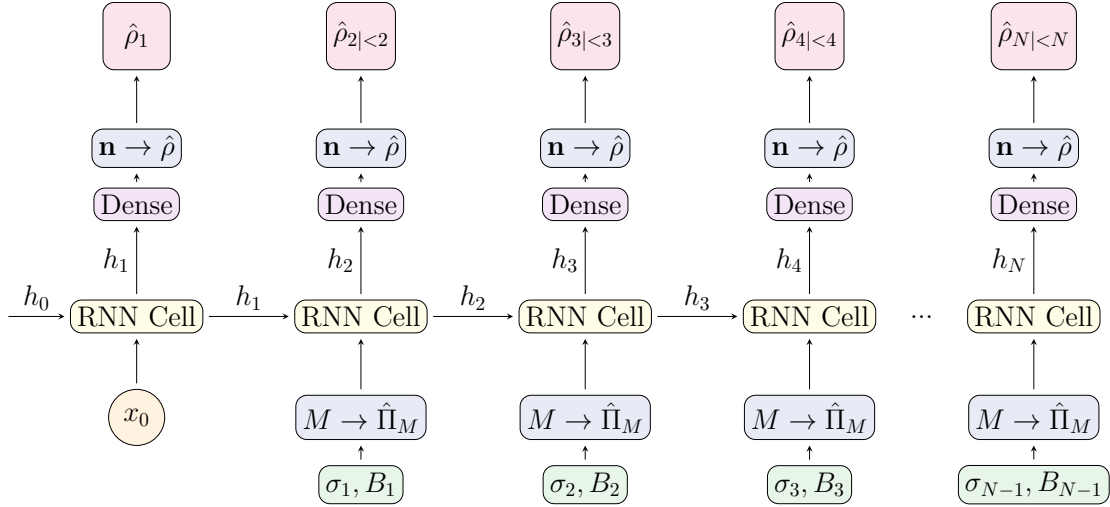


Figure 2.3: **Conditional local density recurrent neural network (CLDRNN)**. h_0 and x_0 are the initial hidden state and initial token respectively required to initiate the RNN, they are typically taken to be the zero vector. The layer $(\mathbf{n} \rightarrow \hat{\rho})$ corresponds to the mapping from the Bloch vector to the local density operator (eq. (2.20)) and the layer $(M \rightarrow \hat{\Pi}_M)$ corresponds to the mapping from a Pauli measurement outcome to the projection operator (eq. (2.21)).

The neural network preserve the autoregressive property as the order of the conditional density matrix decomposition is analogous to the order of the autoregressive sequence. To

obtain the probability of a pauli measurement from the system, we perform the following computation:

$$\begin{aligned} p(\mathbf{M}) &= p(M_1)p(M_2|M_1)...p(M_N|\mathbf{M}_{<N}) \\ &= \prod_i \text{tr} \left(\hat{\Pi}_{\mathbf{M}_i} \hat{\rho}_{A_i|\mathbf{A}_{<i}} \right). \end{aligned} \quad (2.25)$$

Measurements in Pauli basis B can be sampled from the CLDRNN by performing the following procedure:

1. Feed an initial token x_0 to the CLDRNN, the output corresponds to local density of the first qubit $\hat{\rho}_1$.
2. Evaluate the probabilities for Pauli measurement with basis B_1 on the first qubit $p(M_1)$.
3. Sample from the probabilities to obtain the measurement of the first qubit $M_1 = \{\sigma_1, B_1\}$.
4. Repeat step 1 substituting in the outcome of the measurement of the first qubit M_1 for the initial token, the output corresponds to the conditional local density of the second qubit $\hat{\rho}_{2|1}$.
5. Repeat steps 2 and 3 with the conditional local density of the second qubit $\hat{\rho}_{2|1}$.
6. Continue repeating steps 4 and 5 with the next qubit in the sequence until the last qubit is measured.

As we are able to efficiently perform measurements in arbitrary Pauli basis with CLDRNN architecture, we can perform random uniform Pauli measurements on the CLDRNN to generate a dataset larger than the dataset the CLDRNN is trained on. We can perform classical shadows (detailed in section 2.1) on this expanded dataset to obtain estimates for the expectation of physical observables.

2.4 Long-range transverse field Ising model (TFIM) on the trapped-ion quantum processor

We evaluate the methods introduced in sections 2.1 to 2.3 on the ground state of a practical Hamiltonian for the trapped-ion system. Specifically, we consider the 1D long-range anti-ferromagnetic transverse field Ising model (TFIM) with power law decay, with exponent

α , in the interaction strength with distance.

$$\hat{H} = \sum_{i < j} J_{ij} \hat{\sigma}_i^x \hat{\sigma}_j^x + h \sum_i \hat{\sigma}_i^z \quad (2.26)$$

$$J_{ij} \approx + \frac{1}{|i - j|^\alpha}, \alpha \in [0, 3] \quad (2.27)$$

Here, J_{ij} is the interaction strength between the i -th and j -th qubit and h is the transverse field strength. The trapped-ion system is able to simulate long-range TFIM via the Mølmer-Sørensen scheme described in section 1.4.3 [13, 49, 50].

The long-range TFIM has been studied extensively due to their interesting physics in relation to frustration, quantum phase transition, faster information transmission, etc. [45–48, 52]. The 1D long-range anti-ferromagnetic TFIM is known to experience a quantum phase transition near a critical field [46]:

$$h_c \approx \left(1 - \frac{1}{2^{\alpha-1}}\right) \zeta(\alpha), \quad (2.28)$$

where, ζ is the Riemann zeta function.

By performing an adiabatic quantum simulation, the ground state of the long-range TFIM can be prepared to high fidelity [49]. After preparation, random Pauli measurements can be performed via site-dependent single qubit gates followed by a global site-resolved measurement [191], these procedures are described in section 1.4.3. These measurements form the dataset from which we extract information by estimating the expectation of physical observables of the system.

For the remainder of the chapter, we generate data via the exact diagonalization of eq. (2.26) for small systems. We sample a total of $D = 10^5$ random Pauli measurements from the ground state of the system.

2.5 Training procedure for complex RNN wavefunction and CLDRNN

The training of the complex RNN wavefunction and the CLDRNN takes on the same form. With the dataset $\mathcal{D}$ generated from exact diagonalization, we evaluate the Kullback-Leibler

	Complex RNN wavefunction	CLDRNN
N_h	128	64
N_{layers}	2	1
N_{params}	152483	14435
p_{dropout}	0.0	0.4
Epochs	1000	500
RNN cell	GRU	
Batch size	100	
Optimizer	Adam	
Learning rate	3×10^{-4}	

Table 2.1: **The hyperparameters for the training of the complex RNN wavefunction and CLDRNN on the long-range TFIM.** N_h is the size of the hidden state, N_{layers} is the number of RNN cell layers, N_{params} is the total number of trainable parameters and p_{dropout} is the dropout probability during training.

divergence (KLDiv) loss function (eq. (1.24)) for multiple basis:

$$\begin{aligned}
p_{\text{NN}}^B(\sigma) &= p_{\text{NN}}(\mathbf{M}), \\
\mathcal{L}_{\text{KL}} &= \sum_B \text{KLDiv}(p_{\mathcal{D}}^B || p_{\text{NN}}^B) \\
&= \sum_B \sum_{\sigma} p_{\mathcal{D}}^B(\sigma) \log \left(\frac{p_{\mathcal{D}}^B(\sigma)}{p_{\text{NN}}^B(\sigma)} \right) \\
&= \frac{1}{D} \sum_{\mathbf{M} \in \mathcal{D}} [\log(p_{\mathcal{D}}(\mathbf{M})) - \log(p_{\text{NN}}(\mathbf{M}))] \\
&= \mathcal{H}_{\mathcal{D}} - \frac{1}{D} \sum_{\mathbf{M} \in \mathcal{D}} \log(p_{\text{NN}}(\mathbf{M}))
\end{aligned} \tag{2.29}$$

$$\tag{2.30}$$

By removing the constant, entropy of the dataset $\mathcal{H}_{\mathcal{D}}$, the loss is reduced to the negative log-likelihood (NLL) loss.

$$\mathcal{L}_{\text{NLL}} = -\frac{1}{D} \sum_{\mathbf{M} \in \mathcal{D}} \log(p_{\text{NN}}(\mathbf{M})). \tag{2.31}$$

Equations (2.16) and (2.25) gives the functional form of $p_{\text{NN}}(\mathbf{M})$ for the complex RNN wavefunction and CLDRNN respectively. This loss function $\mathcal{C}_{\text{NLL}}$ is minimized during

training using a gradient descent protocol, described in section 1.5.2. The specific optimizer used is described in table 2.1.

We performed a hyperparameter search for both the complex RNN wavefunction and CLDRNN, the optimized values are presented in table 2.1.

The models are implemented with PyTorch [192] and Lightning [193]. The hyperparameter tuning was performed with Ray [194, 195].

2.6 Results

As the complex RNN wavefunction encodes a physical quantum state, evaluation of the exact state is possible, though intractable at large sizes. We compare the fidelity of this RNN NQS with the target state obtained via the exact diagonalization of the Hamiltonian eq. (2.26). The fidelity between a pair of states has the following form:

$$\mathcal{F} = |\langle \Psi_{\text{NN}} | \Psi_{\text{GS}} \rangle|^2. \quad (2.32)$$

The result is consistent with the findings of [164], in the log-log plot of infidelity and training data (fig. 2.4), they are related linearly. The fidelity for the classical shadow and CLDRNN are not well-defined as, in general, they do not capture valid quantum states, pure or mixed.

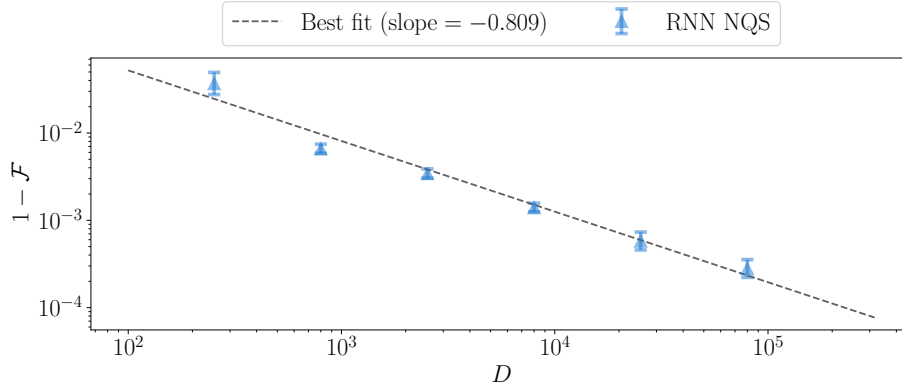


Figure 2.4: **Infidelity of NQS state.** The system parameters are $\alpha = 0.5$ and $h = 0.6$ close to the theoretically predicted critical point $h_c \approx 0.604$ from eq. (2.28). The shaded area is the relative error over 5 realizations of the NQS.

To compare the proposed methods, we study the performance of each method by estimating the error of the full set of Pauli observables.

$$\varepsilon_{\hat{Q}} = \langle \hat{Q} \rangle_{\text{target}} - \langle \hat{Q} \rangle_{\text{method}} \quad (2.33)$$

As previously mentioned, all physical observables can be reproduced by a linear combination of this basis, thus, this serves as a general evaluation of the accuracy of the state produced by the various methods. Figure 2.5 shows that the RNN NQS has a balanced variance for all Pauli operators. In contrast, that the classical shadows and CLDRNN have a weight dependent error profile, higher weight Pauli observables generally behave worse compared with lower weight variants.

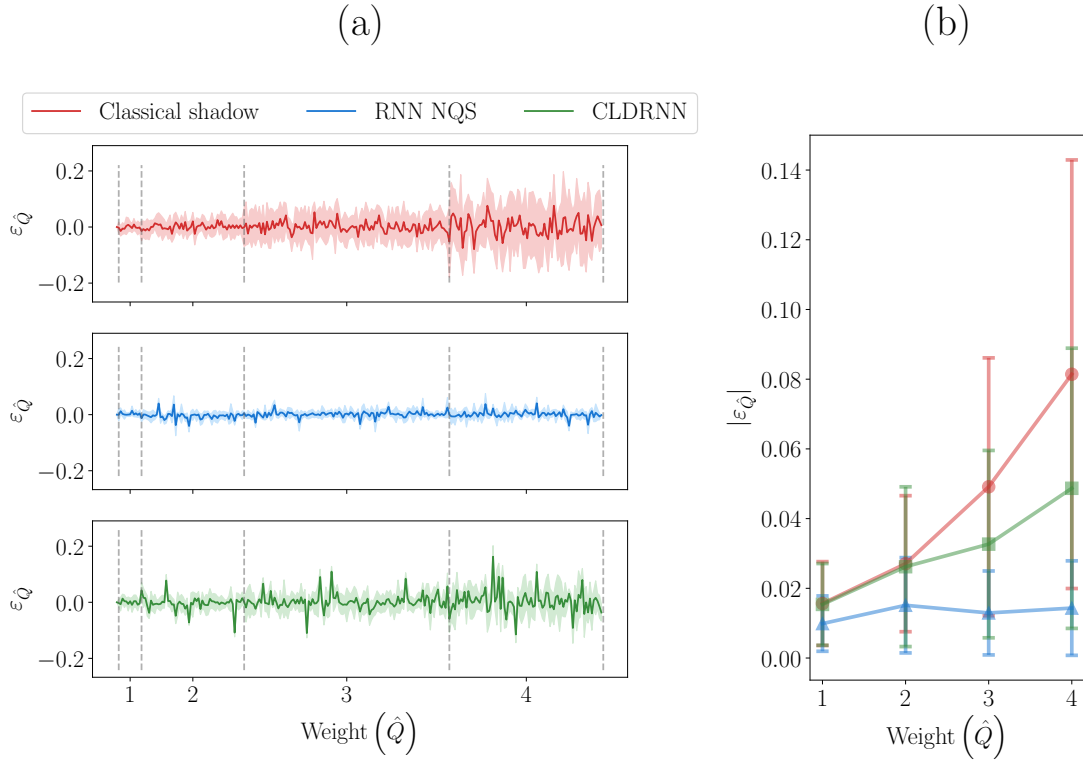


Figure 2.5: **Error in the expectation of Pauli observables for the 4-qubit TFIM ground state.** The methods are evaluated for the system ($\alpha = 0.5$ and $h = 0.6$) with 8000 random Pauli measurements. In (a) and (b), the shaded area and error bars correspond to the standard deviation over 5 realizations of each method.

For classical shadows, this consequence can be derived from Equations (2.10) to (2.12),

where for high-weight Pauli observables, most of the data is discarded and the remaining data reweighted by the function f . This results in the variance of the expectation increasing with weight. The CLDRNN employs the classical shadows for observables estimation, except with an expanded dataset, thus, leading to the same result.

Therefore, NQs are ideal for estimating high-weight Pauli observables with limited data. On average, over all Pauli observables, the CLDRNN outperforms classical shadows for larger weight observables. However, there are Pauli observables for which the CLDRNN exhibit a large bias from the true value. This results in larger errors in energy for the CLDRNN compared to classical shadows seen in fig. 2.6.

Furthermore, we see that there is a correspondence between the energy estimate in fig. 2.6 and the fidelity of the state fig. 2.4. Notice that there exist an extremely data-starved regime ($D \lesssim 200$) where classical shadows outperform the NQS. This arise from the fact that classical shadows form unbiased estimators of the observables, whereas, NQS are known to exhibit a systematic bias [196], this is a variation of the well-known bias-variance tradeoff in the field of machine learning.

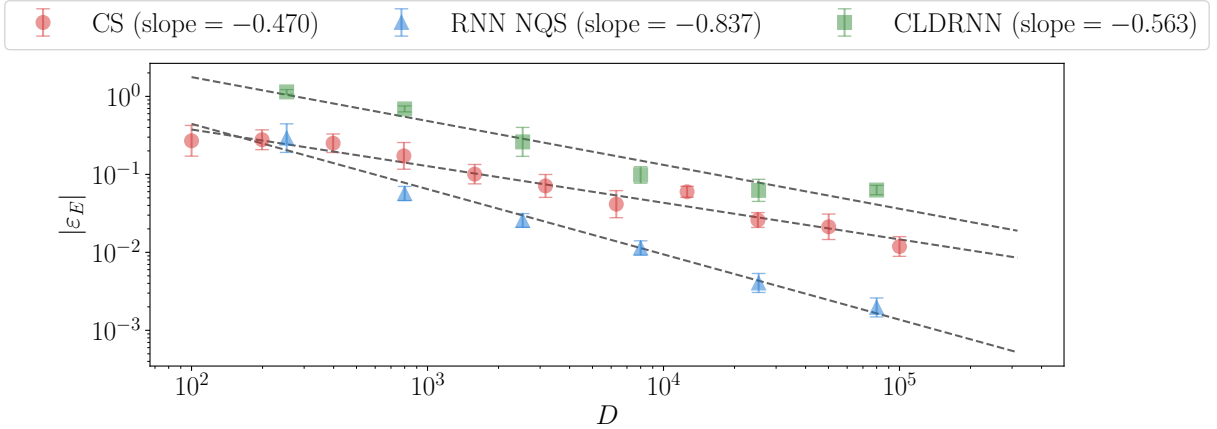


Figure 2.6: **Sampling complexity for TFIM energy.** The system parameters are $\alpha = 0.5$ and $h = 0.6$ near the predicted critical point $h_c \approx 0.604$ from eq. (2.28). The error bars correspond to the relative error taken over 5 different datasets for each size.

Next, we examine the error in the energy estimate as the system size N is scaled. We obtain further evidence of the bias-variance tradeoff. Figure 2.7 shows classical shadows produced an unbiased estimator for the energy, whereas, the machine learning methods include an inductive biases. In contrast, the variance of the NQS is significantly smaller

than that of the classical shadows. The CLDRNN performs significantly worse than the other methods, exhibiting high variance and bias.

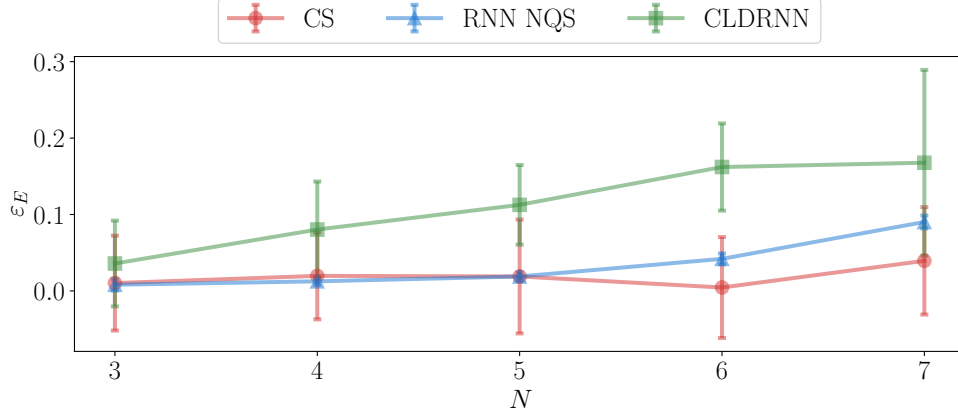


Figure 2.7: **Estimates for TFIM energy for varying system sizes N .** The system parameters are $\alpha = 0.5$ and $h = 0.6$ near the predicted critical point $h_c \approx 0.604$ from eq. (2.28). The error bars correspond to the relative error taken over 5 different datasets for each size.

2.6.1 Computational cost of machine learning models

Figure 2.8 agrees with expectations, the time complexity of training the complex RNN wavefunction scales exponentially with system size whereas the CLDRNN scales linearly with time. At low N , the constant factors in the computation for the models are dominant, as such, the time complexity of training the model does not grow according to the asymptotic scaling.

2.7 Discussion

The CLDRNN enforces the physicality of the state at the local level, however, it does not incorporate any constraints at the global scale. A direct consequence is that the CLDRNN does not strictly represent physical states. The extra space of unphysical states influences the tendency of the CLDRNN to overfit. In contrast, the complex RNN wavefunction is more resilient to overfitting due to the requirement that the state remains physical.

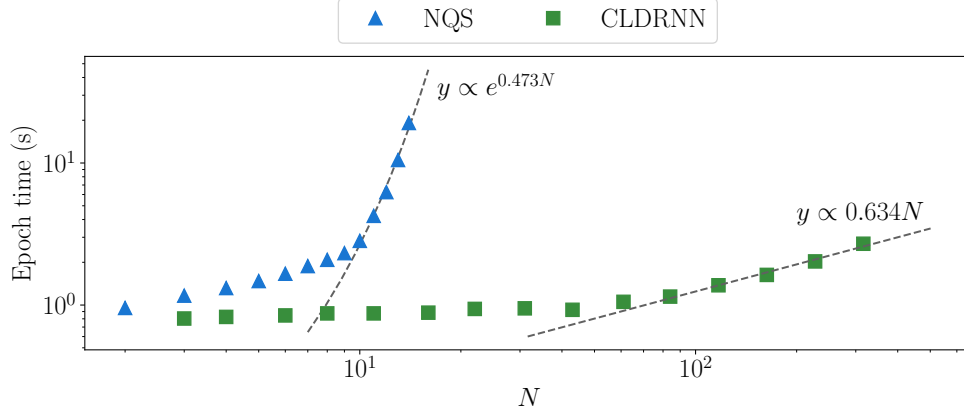


Figure 2.8: **Training time complexity for complex RNN wavefunction and CLDRNN.** The best fit curve and line are plotted for the complex RNN wavefunction and CLDRNN based on the last 5 system sizes evaluated for each architecture.

As previously mentioned, training an NQS, on random Pauli measurements, for a general wavefunction is intractable due to the exponential scaling. However, it is possible to incorporate information known regarding the sign structure to ease this issue. In particular, it is known that the ground state of Hamiltonians, stoquastic in a known basis, are real and positive in said basis. Stoquasticity implies the representation of the Hamiltonian in the basis only consist of non-positive off-diagonal elements [197]. These states are known to have no sign structure, containing only real and positive coefficients, as such, this information can be utilized by removing the phase parameterization of the NQS. This makes the NQS highly efficient as this removes the need for measurements in bases other than the canonical basis and the exponential computation required to rotate the NQS that accompanies it. The probability distribution parameterized by the NQS can straightforwardly be mapped to the quantum wavefunction with the following:

$$\Psi_{\text{NN}}(\sigma) = \sqrt{p_{\text{NN}}(\sigma)}. \quad (2.34)$$

The space of observables that can be efficiently estimated is still limited by the sparse off-diagonal constraints associated with eq. (2.17).

2.8 Conclusion

For the trapped-ion quantum processor, in the data-starved regime, we examine the extraction of information via random Pauli measurements of a variety of protocols. We observe that NQSs perform the best of the proposed methods to obtain an accurate estimate of general observables scaling much better with the expansion of the training dataset. This is especially true for observables with decompositions containing significant contribution from high-weight Pauli observables. However, the computational requirement of NQS for reconstructing general states from the random Pauli basis measurement data is intractable.

Classical shadows are much more computationally efficient for cases where data is abundant or low-weight observables are of interest. In general, there is a tradeoff between the sampling and computational complexities when comparing the methods. Additionally, there is a bias-variance tradeoff when comparing classical shadows with the machine learning methods.

Our proposed architecture of the CLDRNN fails to provide a full alternative for the cases where NQSs are not viable. This is due to the emergence of significant biases in particular Pauli observables leading to potentially worse performance, when compared to classical shadows. However, we do demonstrate that, in general, the CLDRNN outperforms classical shadows for high-weight Pauli observables.

The formulation of a scheme that incorporates constraints on larger subsystems in the CLDRNN, without affecting the capability of efficient local rotations, could improve the performance of the model, providing a middle ground between NQSs and classical shadows. A thorough comparison of the CLDRNN, classical shadows and NQSs with density matrix compatible ansatz [168, 172] for mixed state reconstruction is also warranted.

Furthermore, more efficient basis sets are known for reconstructing pure wavefunctions [198–200]. The corresponding bases rotations are also sparse in nature enabling efficient rotation of the RNN NQS. However, the realization of such bases measurements in near-term quantum devices could be computationally expensive. Optimization of the basis set for both near-term quantum device and efficient NQS basis rotations is a worthy pursuit.

Chapter 3

Autoregressive model path dependence near Ising criticality

From section 1.6.3, we see that autoregressive decomposition breaks down the joint probability distribution into a sequence of conditional probability distribution, we call this sequence the autoregressive path. The autoregressive path is naturally a 1D object. In the previous chapter, we looked at the 1D long-range TFIM model, as such, there is a straight forward choice for the autoregressive path as the dimensions match. For higher-dimensional systems, the choice for the autoregressive path is non-trivial as there is a requirement to map the 1D sequence onto the higher-dimensional space.

The process of mapping a 1D sequence onto a higher-dimensional space corresponds to the concept of space-filling curves [201]. A d -dimensional curve $\gamma(t)$ is defined as a function of the following form:

$$\gamma : [0, 1] \rightarrow [0, 1]^d. \quad (3.1)$$

Space filling curves are a family of curves where $\gamma(t)$ is bijective.

Typically, many-body systems are defined on lattices, as such, these models are discretized in space. As such, there is a need to consider space-filling curves for discrete spaces. This corresponds to the typical construction of space-filling curves where the true space-filling curve is defined as the limit of a set of discrete space-filling curves on finer and finer grid of points. Examples of 2-dimensional (2D) space-filling curves that utilize the above construction include the Hilbert curve [202, 203], Morton curve [204], etc. Some of these space-filling curves have irregular and higher-dimensional generalizations [205–208]. Traditionally, these space-filling curves have been used in resource management for high performance computing and multi-dimensional data storage [209–212].

This chapter consists of material written for publication [1]. In this chapter, we study how the autoregressive path affects the learning behavior of two leading autoregressive architectures, RNNs [164, 176–178] and transformers [2, 179, 213–215], when trained on data that is not intrinsically a 1D sequence. Specifically, we look at the 2D Ising ferromagnet on a square lattice with periodic boundary conditions (PBCs) (shown in fig. 1.1), a paradigmatic model in statistical physics, with Hamiltonian,

$$E = - \sum_{\langle i,j \rangle} \sigma_i \sigma_j, \quad \sigma_i \in \{\pm 1\}. \quad (1.1 \text{ revisited})$$

where $\langle ij \rangle$ denotes nearest-neighbor bonds. Near its well-studied thermal critical point, Ising variables develop long-range correlations that decay as a power law. By mapping lattice sites to sequence elements, we train several autoregressive models on this data, and observe their behavior in regard to learning optimal model parametrizations. We find that training dynamics depends strongly on the arbitrary choice of autoregressive path through the 2D lattice, and discuss possible consequences for improving training behavior of large autoregressive models in the future.

3.1 Autoregressive paths

$$\begin{aligned} p(\mathbf{x}) &= p(\sigma_1)p(\sigma_2|\sigma_1)p(\sigma_3|\sigma_1, \sigma_2)\dots p(\sigma_N|\sigma_1, \sigma_2, \dots, \sigma_{N-1}) \\ &= \prod_{n=1}^N p(\sigma_n|\sigma_{<n}) \end{aligned} \quad (1.32 \text{ revisited})$$

The autoregressive decomposition eq. (1.32) holds for any arbitrary ordering of the variables into a sequence. Therefore, there is a freedom in the assignment of the sequence labels to the sites of a lattice, which generates a path through the system.

For the 2D Ising ferromagnet on a square lattice (eq. (1.1)), we are interested in the multiple ways of constructing paths for 2D systems. Consider an 2D square lattice of size $L \times L = N$, as aforementioned, the path traversing this lattice will be discrete:

$$\mathcal{I}_T = \left\{ t_n = \frac{n}{T} \mid n \in \mathbb{N} \text{ and } n \leq T \right\}, \quad (3.2)$$

$$\gamma : \mathcal{I}_N \rightarrow \mathcal{I}_L \times \mathcal{I}_L. \quad (3.3)$$

Specifically we are interested in paths that exhibit these property: jump-free, locality-preserving, path anisotropy (see section 3.1.2).

3.1.1 Space-filling curves

As previously mentioned, well-known space-filling curves are a natural choice for the autoregressive paths. In this chapter we use the following curves: zigzag curve, snake curve, Morton curve and the Hilbert curve [212].

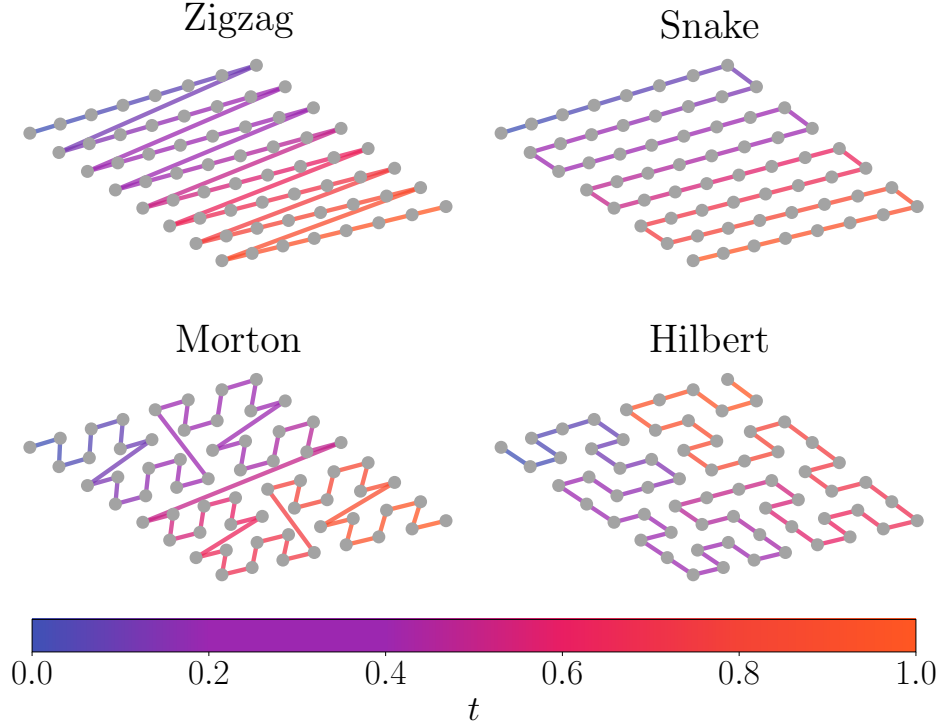


Figure 3.1: **Space-filling curves traversing a 2D square lattice.** The color represents the parameter t which is varied, from 0 to 1, to generate the paths from their respective parametric equations $\gamma(t)$. The diagram is adapted from [1] with permission.

The zigzag curve, also called a raster scan or sawtooth, is one of the simplest space filling curve. The zigzag curve is composed of a set of parallel lines where subsequent lines are connected end to start. The simplicity of the zigzag curve leads it to be the first curve used, as such, it is used in many simulation strategies [176]. Another curve similar to the zigzag curve is the snake curve. For the snake curve, the parallel lines are instead connected by alternating between end to end connections and start to start connections. This curve has been extensively used as the autoregressive path for neural quantum states [177, 216].

One method for generating space-filling curves is with a recursive procedure [201], the Morton curve [204], also known as the Lebesgue curve and the Z -order curve, is one such curve. The Morton curve is generated by starting with a curve in the shape of “ Z ”, the vertices in the curve are then recursively replaced with a “ Z ” and the ends are connected in the same order as before to form the next iteration of the Morton curve. From the generating procedure, we can see a famous property of the Morton curve, its ability to preserve locality. The Morton curve is traditionally used in resource management in high performance computing [209–211].

Another method for generating space-filling curves is with Lindenmayer systems (L-systems) [201, 217]. The Lindenmeyer system is a grammar based description of these curves, where the process of generating iterations of the curve is described by a context-free grammar.

The Hilbert curve is an example of a curve that can be generated from an L-system. Similar to the Morton curve the Hilbert curve is well known for its locality-preserving property [202, 203]. It has been demonstrated that the Hilbert curve has advantages compared to the snake curve for tensor network methods, such as tree tensor networks (TTN) and matrix product states (MPS) [218].

Due to the recursive nature of the generation procedure for the Hilbert and Morton curve, for the $\mathcal{O}$ -th iteration, the linear size of the Hilbert and Morton path is $L = 2^{\mathcal{O}}$. There are algorithms for the generalization of such curves for arbitrary sizes and dimensions [206–208, 219–221].

3.1.2 Path properties

A jump-free path obeys the following:

$$\forall n \leq N, \|\gamma(t_{n+1}) - \gamma(t_n)\| = \frac{1}{L}. \quad (3.4)$$

This states that each step of the curve is of a uniform size equal to the minimum length scale of the discrete space.

A Locality-preserving path implies that the locality measure $\mathcal{C}$ for the path is low. The locality measure takes on the following form [201]:

$$\mathcal{C} = \max \left\{ \frac{\|\gamma(t') - \gamma(t)\|^d}{|t' - t|} \mid \forall t' \neq t \right\}. \quad (3.5)$$

When using the maximum, outliers, corresponding to jumps in the curve, dominate the expression. Therefore, we instead consider the mean:

$$\tilde{\mathcal{C}} = \text{mean} \left\{ \frac{\|\gamma(t') - \gamma(t)\|^d}{|t' - t|} \mid \forall t' \neq t \right\}. \quad (3.6)$$

Anisotropy of the path can be measured with the following:

$$\mathcal{A}_i = \text{mean} \left\{ \frac{|\gamma_i(t') - \gamma_i(t)|}{|t' - t|} \mid \forall t' \neq t \right\}, \quad i \in \{1, \dots, d\} \quad (3.7)$$

$$\mathcal{A}_{ij} = \frac{\mathcal{A}_i}{\mathcal{A}_j}. \quad (3.8)$$

When $\mathcal{A}_{ij} = 1$, the system is isotropic between directions i and j .

Path	Property	Value
Zigzag	Jump-free	No
	Locality measure (mean) $\tilde{\mathcal{C}}$	0.8232
	Anisotropy $\mathcal{A}_{xy}$	2.1075
Snake	Jump-free	Yes
	Locality measure (mean) $\tilde{\mathcal{C}}$	0.7904
	Anisotropy $\mathcal{A}_{xy}$	2.0105
Morton	Jump-free	No
	Locality measure (mean) $\tilde{\mathcal{C}}$	0.8185
	Anisotropy $\mathcal{A}_{xy}$	1.2495
Hilbert	Jump-free	Yes
	Locality measure (mean) $\tilde{\mathcal{C}}$	0.7368
	Anisotropy $\mathcal{A}_{xy}$	0.9269

Table 3.1: **Properties of 2D space-filling curves.** The properties are calculated for a system size of $L = 8$ with PBCs. The path is locality-preserving when the locality measure $\tilde{\mathcal{C}}$ is small and anisotropic when $\mathcal{A}_{xy} = 1$.

The Hilbert path both preserves locality the best and also has the least anisotropy out of the paths considered as can be seen from table 3.1. Figure 3.2 shows that the locality measure of the Morton and Hilbert paths are more stable with system size compared to

the zigzag and snake paths. The anisotropy of the Hilbert and Morton paths decrease with system size, whereas, the anisotropy is increasing with system size for the snake and zigzag path. The zigzag path is not jump-free in general, however, for PBCs, the 2D zigzag curve only has L diagonal jumps at the end of each chain.

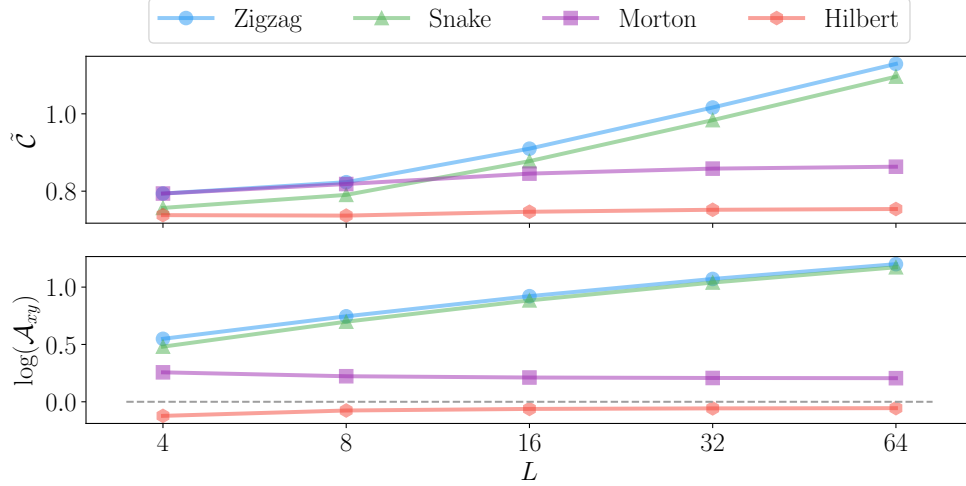


Figure 3.2: Scaling of locality measure $\tilde{C}$ and anisotropy $\mathcal{A}_{xy}$ of autoregressive path with system size N .

3.2 Hyperparameters and training procedure

With the paths proposed in section 3.1, we focus on training two autoregressive architectures: RNNs and transformers with masked self-attention (see section 1.6.3). We train the 1D RNN and transformer using data of the ferromagnetic Ising model on a 2D square lattice with PBCs (eq. (1.1)) obtained via thermal Monte Carlo simulations described in fig. 1.5.

We run the Monte-Carlo simulation for the following system parameters:

$$L \in \{4, 8, 16\}, \quad \beta \in [0.286, 0.667], \quad (3.9)$$

where, β is the inverse temperature. Each dataset consist of $D = 10^4$ uncorrelated samples, 80% of which is used for training and the remainder for validation. We choose the linear sizes $L = 2^O$ for compatibility with the Hilbert and Morton path.

Similar to section 2.5, we train the autoregressive models by minimizing the negative log-likelihood (NLL), $\mathcal{L}^{\text{NLL}}$, of obtaining the dataset. The optimization is performed with Adam [127] and a learning rate of 10^{-3} .

To account for the architectural differences between RNNs and transformers, we tune the hyperparameters with the Asynchronous Successive Halving Algorithm (ASHA) [222] and obtain the optimal parameters for both model architectures outlined in table 3.2.

1D RNN		Transformer	
Parameter	Value	Parameter	Value
N_h	16	N_e	32
N_{layers}	1	N_{FFNN}	512
RNN Cell	GRU	N_{heads}	4
trainable params	3522	N_{blocks}	2
p_{dropout}	0.0	trainable params	75490
		p_{dropout}	0.1

Table 3.2: **Hyperparameters for the training of 1D RNN and transformer on the 2D Ising model.** N_h and N_{layers} are the hidden state size and number of RNN cell layers used in the 1D RNN respectively. N_e is the embedding size, N_{FFNN} is the dimension of the FFNN, N_{heads} is the number of heads used in MHA N_{blocks} is the number of transformer blocks used in the transformer architecture. p_{dropout} is the probability of dropout during training.

The models are implemented with PyTorch [192] and Lightning [193]. The hyperparameter tuning was performed with Ray [194, 195].

3.3 Results

We first study the performance of the models near criticality at $\beta = 0.435$. The exact critical point of the 2D Ising model is at $\beta_c = \ln(1 + \sqrt{2})/2 \approx 0.4407$ determined in Onsager’s solution [38]. As can be seen in fig. 3.3, the training performance of each model depends strongly on the autoregressive path formulation. The paths that perform best for the RNN also perform best for the transformer, however the transformer loss is minimized at a significantly faster rate (in units of epochs). In particular, the transformer is able to

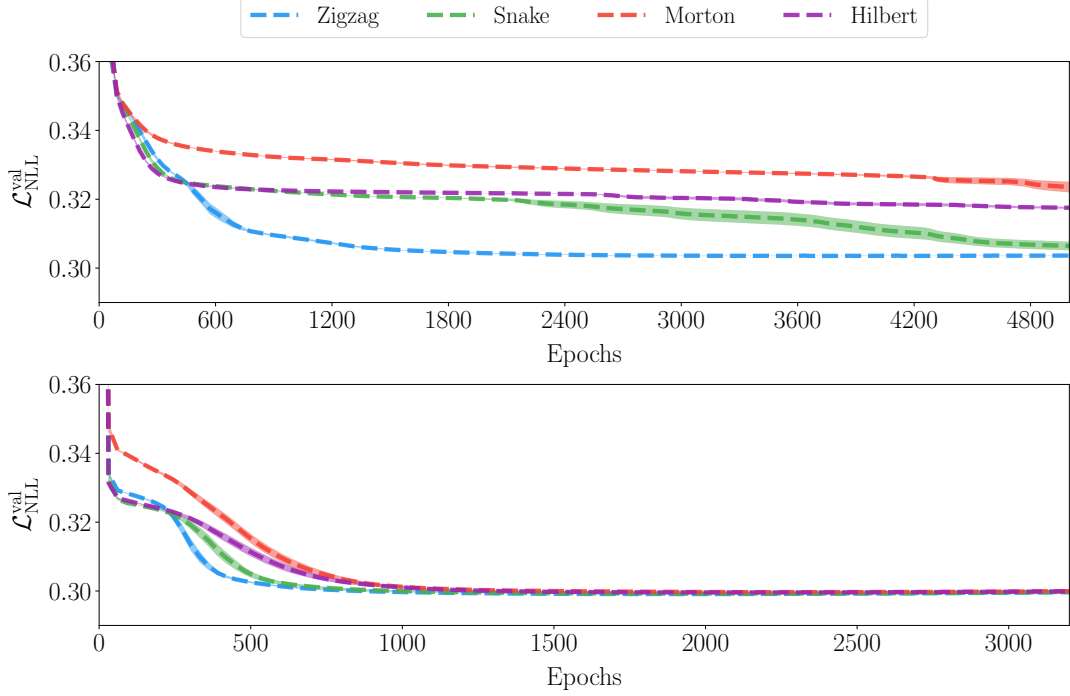


Figure 3.3: **Training results for the 2D Ising model near criticality at $\beta = 0.435$ with linear size $L = 8$ for 1D RNN (top) and transformer (bottom).** The shaded area represents the standard error over 10 independent training runs. The diagram is adapted from [1] with permission.

converge relatively rapidly regardless of the autoregressive path, compared to the 1D RNN which requires a large number of epochs in order for the sub-optimal paths to converge.

In fig. 3.4, we study the two-point spin-spin correlation function in the transformer as it is trained, specifically its anisotropy. The two-point spin-spin correlation function $\mathcal{G}$ has the following form:

$$\mathcal{G}(\Delta x, \Delta y) = \frac{1}{N} \sum_{x,y} (\langle \sigma_{x,y} \sigma_{x+\Delta x, y+\Delta y} \rangle - \langle \sigma_{x,y} \rangle \langle \sigma_{x+\Delta x, y+\Delta y} \rangle), \quad (3.10)$$

$$\Gamma = \frac{\|\mathcal{G} - \text{Rotate}(\mathcal{G})\|}{\|\mathcal{G}\|}, \quad (3.11)$$

where, $\sigma_{x,y}$ is the spin at lattice site (x, y) , Γ is a measure of the anisotropy in the correlations between the x and y directions and $\text{Rotate}(\mathcal{G})$ is a rotation of the correlation matrix

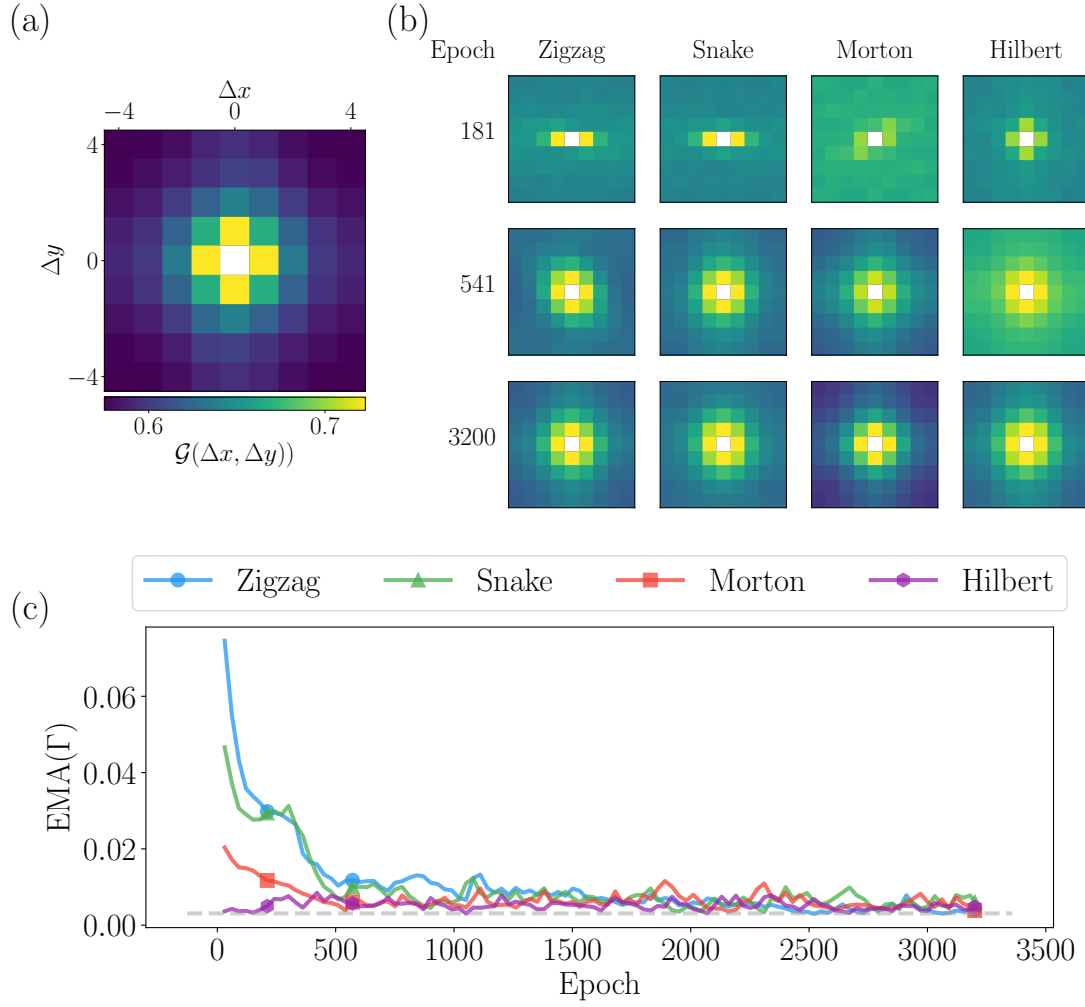


Figure 3.4: **Evolution of the two-point spin-spin correlations $\mathcal{G}(\Delta x, \Delta y)$ during the training of the transformer for the 2D Ising model near criticality at $\beta = 0.435$ with linear size $L = 8$.** (a) Actual spin-spin correlation of the system, (b) spin-spin correlation of the transformer and (c) anisotropy Γ in the spin-spin correlations of the transformer. EMA($\star$) is the exponential moving average used to smooth the plot.

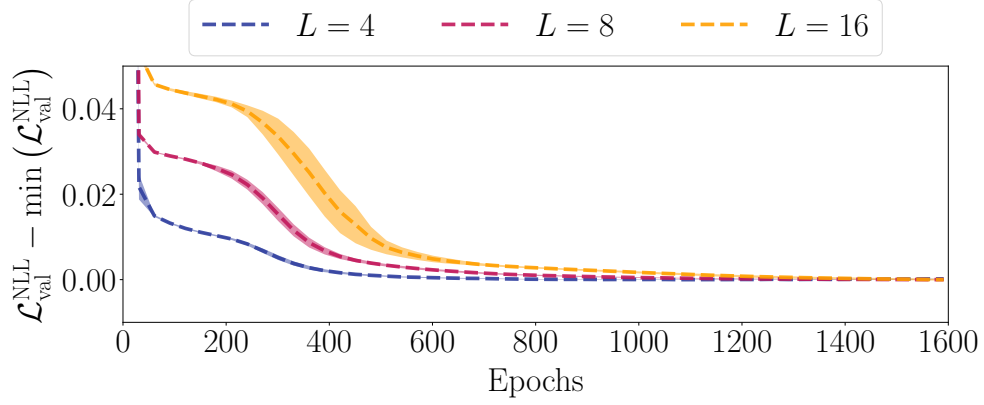


Figure 3.5: **Training of the transformer with the zigzag path on different system sizes.** The transformer is trained on different sizes $L = 4, 8$ and 16 near criticality ($\beta = 0.435$). The shaded area represents the standard error over 10 independent training runs. The diagram is taken from [1] with permission.

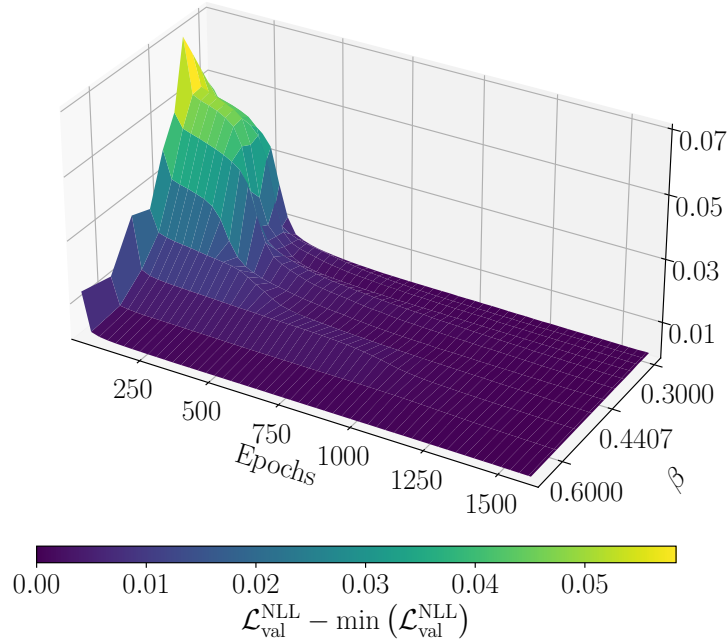


Figure 3.6: **Training of the transformer with the zigzag path at different temperatures for the 2D Ising model.** The transformer is trained on different temperatures, β between 0.286 and 0.667 , across the phase transition for $L = 8$. The diagram is taken from [1] with permission.

by 90 degrees. The square lattice is isotropic with no preferred direction, therefore, the correlations within the system should reflect this property. Figure 3.4(b) shows that during the early-phase training of the autoregressive model with the zigzag and snake path, we observe an anisotropy in the captured correlations. In contrast, the correlations in models with the Hilbert and Morton paths are near-isotropic. This behavior holds for both the 1D RNN and the transformer. Comparing the early learning of the transformer (before 300 epochs in fig. 3.3) with the anisotropy of the spin-spin correlation in the transformer (shown in fig. 3.4(c)), we see a correspondence. Our results indicate that the slowdown in learning early on for the transformer) with the zigzag and snake paths is due to the anisotropy. When the model resolves this anisotropy, it is able to quickly reach the optimal loss.

Finally, it is interesting to observe that even with the anisotropic learning of correlations in the zigzag and snake paths, models with these paths converge faster than the other paths for the case of the Ising model. This is in contrast to the results with TTN and MPS [218], where the Hilbert path was demonstrated to be advantageous.

As we scale the transformer, we see that the anisotropic learning of correlations for the zigzag path is consistent across sizes $L = 4, 8$ and 16 . We note that the number of epochs required to resolve the anisotropy does not significantly depend on the size of system for the transformer architecture (as shown in fig. 3.5).

Finally, we extended our training regime to include a broad range of temperatures across the phase transition of the 2D Ising model. Due to the algebraic decay of correlations at criticality, training near criticality is the most difficult, supported by fig. 3.6. In fig. 3.6, we observe that the initial plateau due to the anisotropic learning of correlations in the models with the zigzag path peaks near the transition and disappears far into the ordered and disordered phases. This provides evidence for the importance of studying the performance of such models across the phase transitions.

3.4 Conclusion

Our results show that the selection of the autoregressive path impacts the learning characteristics both recurrent neural networks (RNNs) and transformers. In the case of a 1D RNN, the locality-preserving paths (Hilbert and Morton) are clearly the worst performers, failing to converge to a good loss value even after significant epochs of training. This result is surprising, since locality-preserving paths have recently been shown to give superior performance in 1D Matrix Product State (MPS) simulations [218]. In contrast, for the transformer the final result of the models' training is essentially independent of the autoregressive path. However, the locality-preserving paths take more epochs to train than the

zigzag and snake paths, converging in the same order (zigzag, snake, Hilbert then Morton) as observed in the RNN. Our results also show that the transformer is able to learn correlations at a significantly faster rate (per epoch) compared to the 1D RNN. Overall, for the Ising model on a 2D square lattice with periodic boundary conditions, we conclude that the zigzag path performs best out of the various autoregressive paths we tested. We conjecture that in the comparison between the zigzag and snake path, the zigzag path performs better due to the periodic boundary condition of the model in study. The periodic boundary contains connections that wrap around the boundary, present in the zigzag path but absent in the snake path. It would be interesting to repeat our study for models of other geometries, dimensions, and interacting Hamiltonians.

This work reveals the need to study more closely the effect of autoregressive path choice in the training of modern language models. While our results are for relatively vanilla architectures, in recent years there have been multiple proposals of modified transformers with reduced scaling in sequence length [223–228]. In certain cases, such models have been demonstrated to outperform the vanilla transformer with significantly fewer resources. It would be interesting to examine the training efficiency of these models with varying autoregressive paths. Perhaps the combination of modern transformer architecture with a variational principle which optimizes the autoregressive path will lead to increased performance on the expensive training task which faces all large language models.

Chapter 4

RydbergGPT: A large language model for Rydberg atom array physics

In the previous chapters, models were trained to reconstruct a particular state, this is inefficient if we were trying to learn a family of states, parameterized by some system parameters $\mathbf{y}$, as the model would need to be retrained for each value of the system parameters. One way to handle this is with transfer learning [229], where, instead of training a newly initialized network with random weights, the weights of a trained network, preferably where the system parameters $\mathbf{y}$ are close, is used as a starting point for the optimization. However, this still requires adjustments to the network.

In this chapter, we consider a encoder-decoder attention-based transformer architecture that encodes a state conditioned on the system parameters $\mathbf{y}$. The family of states of interest to us is the ground states of parameterized Hamiltonian. These systems exhibit phase transitions as the system parameters are modified, as such, different regions in the phase diagram exhibit different properties. Interesting physics occurs in the transition between phases with regards to the emergence of order, divergences in correlations and other quantities, universality between different models etc. Our architecture, takes as input the system parameters $\mathbf{y}$ and outputs a model representing the ground state of the system as a probability distribution. We apply our architecture to the governing Hamiltonian of the Rydberg atom array quantum processor. The transformer is trained on a variety of different system parameters encompassing a pair of phases, the disordered and anti-ferromagnetic checkerboard phase. We demonstrate that the transformer is able

to interpolate between parameters and acquire a reasonable prediction for the properties corresponding to the ground state of Hamiltonians with system parameters outside of the training regime. Our results give optimism that attention-based transformer models are well-positioned to provide a powerful approach for predicting the properties of quantum many-body systems. This chapter consists of material written for publication [2].

4.1 Rydberg atom array physics

The Rydberg atom array quantum processor, introduced in section 1.4.4, is a powerful platform for quantum information processing. The native Hamiltonian of the Rydberg atom array are as follows:

$$\hat{H} = \sum_{i < j} C_6 V_{ij} \hat{n}_i \hat{n}_j - \delta \sum_i \hat{n}_i - \frac{\Omega}{2} \sum_i \hat{\sigma}_i^x, \quad (1.15 \text{ revisited})$$

$$C_6 = \Omega \left(\frac{R_b}{a} \right)^6, \quad V_{ij} = \left(\frac{a}{\|\mathbf{r}_i - \mathbf{r}_j\|} \right)^6. \quad (1.16 \text{ revisited})$$

where $\hat{\sigma}_i^x = |g\rangle_i \langle r|_i + |r\rangle_i \langle g|_i$, the occupation number operator $\hat{n}_i = \frac{1}{2} (\hat{\sigma}_i + \mathbb{1}) = |r\rangle_i \langle r|_i$ and $\hat{\sigma}_i = |r\rangle_i \langle r|_i - |g\rangle_i \langle g|_i$.

The Rydberg Hamiltonian is naturally a long-range spin model as the Rydberg blockade term behaves as a Van der Waals type interaction, i.e. a power law interaction that decays in distance with an exponent of $\alpha = 6$. This interaction is intrinsic to the Rydberg state, as such, it cannot be turned off. To adjust the interaction term, the atoms are assembled into specific geometries that give rise to the target interaction. Here, we are interested in the 2D square lattice with lattice spacing a . This geometry of the atoms is encoded by a symmetric matrix $\mathbf{V}$ in eq. (1.16). Additionally, the Rydberg Hamiltonian depends on the parameters of the Raman beams, the detuning δ and Rabi frequency Ω . The Rabi frequency Ω is typically taken to be the overall energy scale for the system. From the lattice spacing a and the energy scale Ω , we define a characteristic radius of Rydberg blockade R_b , in which simultaneous excitations are penalized, with eq. (1.16) [111, 113]. For our simulations, we consider the system to be in thermal equilibrium with finite temperature $T = 1/\beta$.

The Rydberg Hamiltonian contains rich physics in its phase diagram with multiple distinct phase transitions [11, 14, 92, 230–232] to explore. Another nice property of the Rydberg Hamiltonian is its stoquasticity, resulting in a real and positive ground state wavefunction

[77, 92, 230, 233]. Consequently, not only can measurements of the system be simulated efficiently via QMC, projective measurements in the occupation basis are sufficient for the reconstruction of the ground state with NQSs.

4.2 Physical observables

The physical observables of interest in the Rydberg Hamiltonian consist of the:

- Staggered magnetization $\langle \hat{\sigma}^{stag} \rangle$
- x -magnetization $\langle \hat{\sigma}^x \rangle$
- Energy E

Diagonal observables, such as the staggered magnetization, can be directly computed from samples taken from the NQS.

$$\langle \hat{\sigma}^{stag} \rangle \approx \frac{1}{D} \sum_{\sigma \sim p(\sigma)} \left| \sum_{i=1}^N (-1)^i \left(n_i(\sigma) - \frac{1}{2} \right) \right|, \quad (4.1)$$

where, i runs over all $N = L \times L$ atoms for the snake path through the 2D square lattice. The staggered magnetization is the order parameter for the disorder-to-checkerboard quantum phase transition under investigation[234].

For estimating the expectation of off-diagonal observables in the NQS, we utilize the local estimators introduced in eq. (2.17). The calculation of the spatially averaged x -magnetization $\hat{\sigma}^x$ is as follows:

$$\langle \hat{\sigma}^x \rangle \approx \frac{1}{D} \sum_{\sigma \sim p_\lambda(\sigma)} \sum_{\sigma' \in \text{SSF}(\sigma)} \frac{\Psi_\lambda(\sigma')}{\Psi_\lambda(\sigma)}, \quad (4.2)$$

where $\text{SSF}(\sigma)$ is the set of configurations that are connected to σ by a single spin flip.

From eq. (1.16), the energy of the system is a combination of the x -magnetization and a diagonal observable. Therefore, estimating the energy is trivially,

$$\langle E \rangle = \sum_{i < j} C_6 V_{ij} \langle \hat{n}_i \hat{n}_j \rangle - \delta \langle \hat{n} \rangle - \frac{\Omega}{2} \langle \hat{\sigma}^x \rangle, \quad (4.3)$$

where, the correlation $\langle \hat{n}_i \hat{n}_j \rangle$ and density $\langle \hat{n} \rangle$ can be directly estimated from samples.

4.3 Encoder-decoder transformer architecture

The system parameters of the Rydberg Hamiltonian, eq. (1.16), can be represented as a fully connected graph, G , consisting of vertices $\mathcal{V}$ and edges $\mathcal{E}$. The vertex attributes correspond to the global and local system parameters, whereas, the edge attributes correspond to the interactions between the spins. We encode global system parameters as vertex attributes such that they can be passed into a typical graph neural network. The following set of equations establish the mapping of the non-dimensionalized system parameters into a graph representation:

$$G = \{\mathcal{V}, \mathcal{E}\}, \quad (4.4)$$

$$\mathbf{y}_{\mathcal{V}} = \left\{ \Omega, \frac{\delta_{\mathcal{V}}}{\Omega}, \frac{R_b}{a}, \beta\Omega \right\}, \quad \mathbf{y}_{\mathcal{E}} = \{V_{\mathcal{E}}\}. \quad (4.5)$$

Fully connected graphs naturally scales quadratically with the system size N due to the inclusion of all possible edges. We choose to process the graph with a graph convolutional neural network (GCN) [235, 236], where information of the edge attributes are incorporated into the nodes. We consider composing multiple layers of GCN, after each of the layers, we apply a layer norm [237]. The output of the last GCN layer correspond to sequence of the transformed node attributes $\mathbf{s}$, as shown in fig. 4.1. This fully encodes the system variables into a sequence that scales linearly with system size.

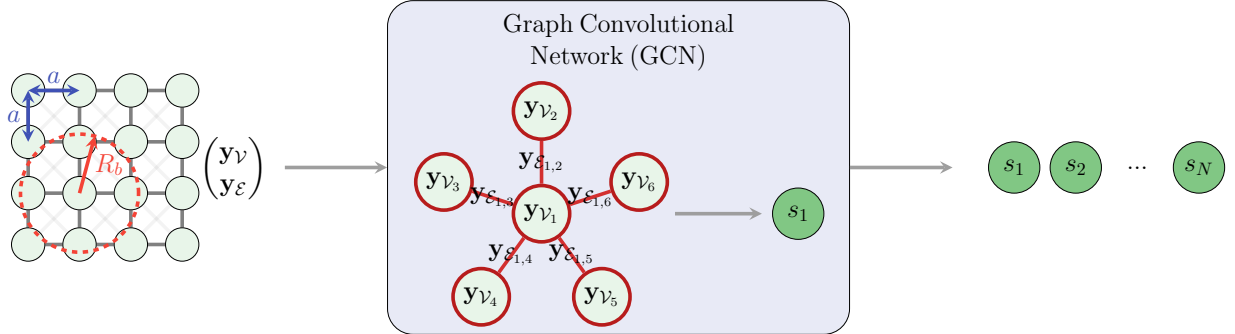


Figure 4.1: **Encoding of the system parameters via a graph convolutional neural (GCN).** The GCN systematically considers each node and the edges connected to it. It uses the edge attributes and neighbor node attributes to update the state of the node.

Our goal is to parameterize, with parameters λ , the probability distributions of the spins conditioned on the encoded system variables $p_{\lambda}(\sigma|\mathbf{s}(G))$. For this purpose, we propose the

use of the vanilla encoder-decoder transformer [149], shown in fig. 4.2. The transformer is a universal sequence-to-sequence function approximators and, granted sufficient tunable parameters λ , can encode arbitrarily complex functions [238].

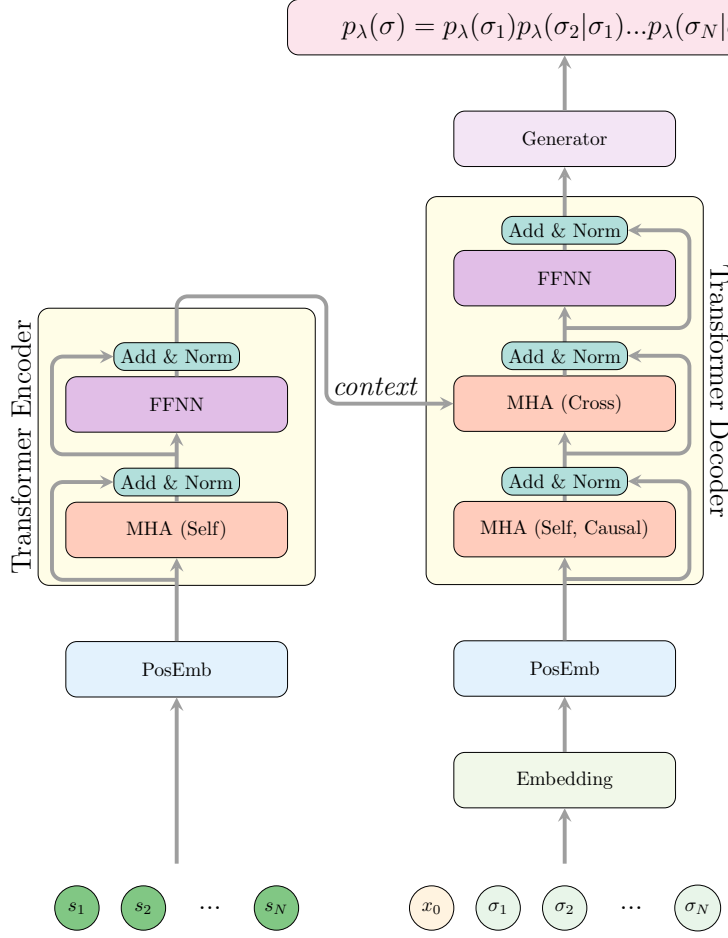


Figure 4.2: **Encoder-decoder transformer architecture.** The diagram is taken from [2] with permission.

The transformer encoder is used to further capture correlations by passing the encoded system variables $\mathbf{s}$ through the self multi-head attention (MHA) mechanism and feed forward layers to obtain a context vector. The context vector is then fed to the transformer decoder via a cross MHA with the intermediate state of the decoder.

We outfit the transformer decoder with the autoregressive property by instituting a causal mask on the self-attention mechanism. As a result, the transformer decoder represent a

joint probability distribution and its outputs are the conditional probability distributions from the autoregressive decomposition. This enables IID samples to be drawn efficiently from the transformer decoder. The joint probability distribution represented by the transformer decoder corresponds to a pure quantum wavefunction conditioned on the system parameters G after the application of the Born rule $p_\lambda(\sigma|G) = |\Psi_\lambda(\sigma|G)|^2$. As previously mentioned, as there is no sign structure for the ground states of the Rydberg Hamiltonian, $\Psi_\lambda(\sigma|G) = \sqrt{p_\lambda(\sigma|G)}$ [113, 177].

Sampling from our architecture involves processing the system parameters of interest with the GCN and transformer encoder to obtain a context vector. Following that, the decoder, conditioned on this context, initializes with a starting token and samples the state of the first spin in the sequence. Then the subsequent spins are iteratively sampled conditioned on all spins preceding it. The lattice of atoms is systematically traversed with the snake path convention, as shown in fig. 3.1. Our choice of the snake path is influenced by the OBCs in the Rydberg Hamiltonian and the fact that the snake path maps the 2D checkerboard pattern to the 1D alternating pattern regardless of size.

A powerful property of sequential models is the ability to unroll the model for arbitrary sequence lengths. This enables us to investigate our architecture’s ability to generalize in system size, i.e. accurately predict the behavior of the Rydberg atom array model as the system is scaled up and down.

The models are implemented with PyTorch [192], Lightning [193] and PyTorch Geometric [239].

4.4 Training transformer models

We follow a data-driven training procedure, where the dataset $\mathcal{D}$ is composed of pairs of system parameters G and samples σ . Our samples are derived from simulated Rydberg occupation measurements, produced via stochastic series expansion (SSE) quantum Monte-Carlo (QMC) method described in [77]. We produce data with QMC for a wide range of settings:

$$\begin{aligned} \beta\Omega &\in \{0.5, 1, 2, 4, 8, 16\}, \quad \frac{R_b}{a} = 1.15, \\ \frac{\delta}{\Omega} &\in [-0.364, 3.173], \quad L \in \{5, 6, 11, 12, 15, 16\} \end{aligned} \tag{4.6}$$

Without loss of generality, we take $\Omega = 1$ henceforth. For each configuration, we generate a dataset containing $D = 10^5$ samples, the total number of tokens is outlined in table 4.1.

Lattice size, L	5	6	11	12	15	16
Tokens (10^8)	1.5	2.16	7.26	8.64	13.5	15.4

Table 4.1: **Number of tokens, in units of 10^8 , contained in our dataset for various sizes L .** A token corresponds to the measurement outcome of a single Rydberg qubit.

The datasets are selected to explore δ/Ω values near a quantum phase transition in the 2D square lattice Rydberg atom arrays, which occurs at approximately $\delta/\Omega \approx 1.1$ [77]. The dataset is available at [240].

Similar to section 2.5, we train the transformer architecture by minimizing the negative log-likelihood (NLL), $\mathcal{L}^{\text{NLL}}$, of obtaining the dataset. We initialize each model using the method introduced by Glorot *et al.* [241] The model is then trained with the AdamW optimizer [242], an extension of stochastic gradient descent with adaptive learning rate and weight regularization, in combination with a cosine annealing warm restart learning rate scheduler [243]. Details for the architecture and training hyperparameters can be found in table 4.2.

RydbergGPT		Training	
Parameter	Value	Parameter	Value
N_h^{GCN}	64	Optimizer	AdamW
$N_{\text{layers}}^{\text{GCN}}$	2	Batch size	1024
N_e	32	Learning rate	1×10^{-3}
N_{FFNN}	128	Learning rate schedule	Cosine annealing warm start
N_{heads}	8	T_0	1
$N_{\text{blocks}}^{\text{Enc}}$	1	T_{mult}	2
$N_{\text{blocks}}^{\text{Dec}}$	3	η_{min}	1×10^{-5}
trainable params	66562	p_{dropout}	0.1

Table 4.2: **The hyperparameters for the transformer encoder-decoder architecture and optimizers.** The GCN has parameters N_h^{GCN} and $N_{\text{layers}}^{\text{GCN}}$, the hidden dimension size and number of layers in the GCN. The transformer parameters include the embedding size N_e , the number of heads for the MHA N_{heads} , the dimension of the feedforward network layer N_{FFNN} and the number of encoder and decoder blocks $N_{\text{blocks}}^{\text{Enc}}$ and $N_{\text{blocks}}^{\text{Dec}}$ respectively.

Three separate models were trained on different subsets of the simulated data. Specifically, the models differ in the sets of linear size they were trained on as detailed in table 4.3. For each size, the models were trained on all values of δ/Ω and $\beta\Omega$. Notice, as we generated the same number of samples for each lattice size, the larger lattice sizes have more tokens compared to the smaller ones. Each model was trained with a dataset on an A100 GPU for 85 hours, during which they processed on the order of 10^{10} tokens.

Model	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$
Linear Sizes L	5	5, 6	5, 6, 11, 12
Epochs	116	40	17
Tokens in dataset (10^8)	3.66	19.6	48.4
Tokens processed (10^{10})	4.25	7.82	8.23
Training time	85h on an A100 GPU		

Table 4.3: **Details on the training of multiple RydbergGPT models.** The table includes the linear sizes used to train the models, the number of epochs trained, the amount of tokens in the training dataset, in units of 10^8 , and the total number of tokens processed during training, in units of 10^{10} .

4.5 Inference with trained transformer models

We performed inference with the trained model to study the disordered-to-checkerboard phase transition of the Rydberg atom array, governed by the dimensionless detuning δ/Ω . To probe the physical observables of interest listed in section 4.2 at different system parameters G , we use G to perform the sampling procedure described in fig. 4.2. The samples are then used to obtain predictions of physical observables. We compare our model’s predictions with estimates obtained from QMC simulations [77]. For Rydberg atom arrays, QMC is able to obtain an estimate of the observables that can be considered exact, within statistical errors. In showing that our model’s predictions of these observables are in good agreement with the QMC values, we illustrate the efficacy of data-driven training as a technique for predicting Rydberg atom array measurement outcomes for ground states. Further, we explore our trained models’ ability to generalize away from the ground state and also to system sizes not represented in the training data.

The computational cost of sampling in our implementation scales as $\mathcal{O}(N^3)$, where the

system size $N = L \times L$. However, this complexity can theoretically be reduced to $\mathcal{O}(N^2)$ by caching the appropriate intermediate values in the attention calculation.

4.5.1 Results

It is important to emphasize that the estimates of off-diagonal observables are appropriate only when the output of the decoder can be interpreted as the normalized wavefunction amplitude, $\Psi_\lambda(\sigma|G) = \sqrt{p_\lambda(\sigma|G)}$. In particular, this correspondence is valid when the system is in a pure state and the wavefunction can be assumed to be real and positive in the occupation basis, as is the case for the Rydberg Hamiltonian in its $T = 0$ ground state. Importantly, the use of the decoder output as a wave function amplitude is no longer valid for finite temperature states. While the above estimates can still be computed, $\Psi_\lambda(\sigma)$ is not able to capture the physics of the thermal state, as is demonstrated in our results which follow.

We now demonstrate the results of the trained models across parameter regimes and observables. In fig. 4.3 we illustrate the performance of model $\mathcal{M}_1$ on the three observables introduced in section 4.2 as a function of δ/Ω , for $R_b/a = 1.15, \beta\Omega = 16.0$. The estimators are calculated over $N_s = 10^5$ samples. The predictions are benchmarked against the QMC observables calculated from 10^5 decorrelated samples, which we take as the ground truth. At $\beta\Omega = 16$, the temperature is sufficiently low for the Rydberg atom array to be in its ground state to a good approximation. The model shows good agreement with the QMC estimates for the energy and staggered magnetization. As for the x -magnetization, we observe a slight deviation from the QMC estimates in the tails. We conclude that the trained model is able to generalize well to dimensionless detunings, δ/Ω , that lie outside of the training data set, when the system is restricted to the ground state.

Next, we investigate how well the model generalizes away from the ground state. In fig. 4.4 we illustrate predictions made by model $\mathcal{M}_1$ with the same strategy, except temperature $T = 1/\beta$ is varied instead of the detuning. We see a strong disagreement between the exact observables and the model-obtained estimates at high temperatures. In our architecture, the decoder represents a probability distribution over configurations in the Rydberg occupation basis, which can be mapped one-to-one onto a positive wavefunction of a pure state. As mentioned, the $T = 0$ ground state of the Rydberg Hamiltonian section 4.1 is known to take such a form, and thus the model performs well for the ground state, e.g. low temperatures. However, as the temperature is increased, the system becomes a well-mixed Gibbs state, which can no longer be accurately represented with a positive wavefunction. As such, the output of the decoder is incapable of accurately representing

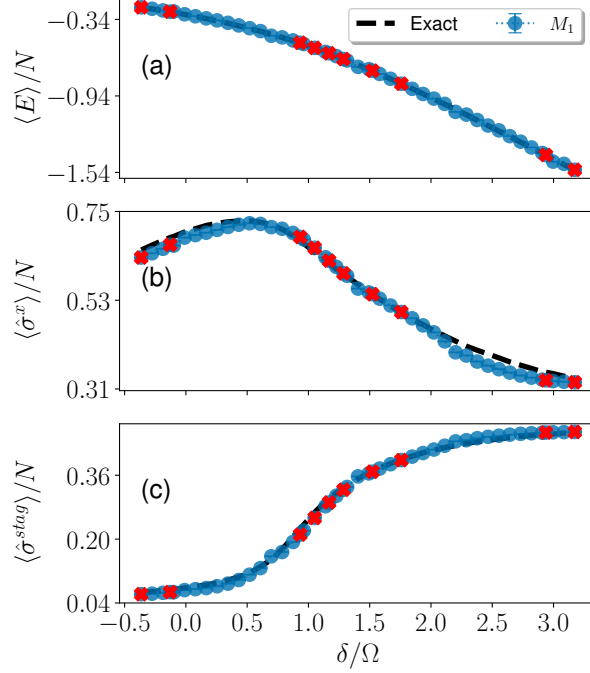


Figure 4.3: **Model predictions of multiple observables across the disordered-to-checkerboard phase transition, which is governed by the dimensionless detuning, δ/Ω .** The observables include the site-averaged energy, x -magnetization and staggered magnetization. Here, blue points correspond to observables estimated from projective measurements sampled from the trained model (i.e., out-of-distribution), red points indicate parameters contained within the training dataset (i.e., in-distribution), and dotted black lines correspond to the exact observables. The diagram is taken from [2] with permission.

this state. The model is merely approximating a probability distribution that results in identical occupation-basis measurement likelihoods. This results in the incorrect estimate of off-diagonal observables, which require the wave function interpretation of the decoder output $\Psi_\lambda(\sigma) = \sqrt{p_\lambda(\sigma)}$, as in eq. (4.2). To appropriately capture such high temperature states, one might employ more sophisticated methods such as a complex density matrix [244] or minimally entangled typical thermal states (METTS) [245, 246].

This is particularly evident in fig. 4.4 at the limit of large temperature. Here, the system tends to the fully-mixed state where all states are equi-probable. The positive wavefunction that shares the same computational basis probability as this state is the equi-superposition of all states. This state corresponds to the maximum eigenstate of the x -magnetization,

leading to the divergence in fig. 4.4 between the exact QMC and the trained model estimators at high temperatures, where $\langle \hat{\sigma}^x \rangle$ becomes significantly over-estimated. This is further supported by the fact that the high-temperature estimates of the staggered magnetization are somewhat accurate. Again, this behavior is due to the fact that this diagonal estimator relies only on the computational basis probabilities with which certain configurations are sampled.

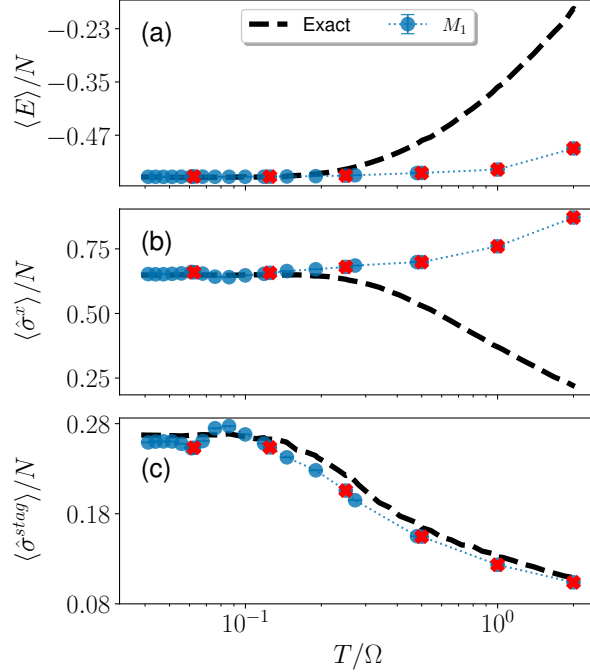


Figure 4.4: **Model predictions of multiple observables as the dimensionless temperature T/Ω is increased.** The observables include the site-averaged energy, x -magnetization and staggered magnetization. Again, blue points correspond to observables estimated from projective measurements sampled from the trained model (i.e., out-of-distribution), red points indicate parameters contained within the training dataset (i.e., in-distribution), and dotted black lines correspond to the exact observables. The diagram is taken from [2] with permission.

Finally, we assess the ability of our transformer to generalize to larger system sizes beyond those included in the training set. For this we use all three trained models, $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$. Figure 4.5 presents the energy evaluations from each of our three trained models, averaged over 10^4 samples. These models were trained on consecutively larger datasets

as previously described in table 4.3. We first observe that the model $\mathcal{M}_1$ trained on the smallest dataset, containing only data from lattice sizes $L = 5, 6$, captures the $L = 5$ energy accurately but struggles to generalize to larger system sizes. Interestingly, the models trained on more extensive datasets do not accurately capture the properties of their respective training sets, with the energy estimates deviating significantly from expected values.

These results illustrate how physical estimators are affected by bias in the training datasets. As shown in table 4.1, we always accumulate 10^5 samples for each grid size. Therefore, the number of training tokens or measurements varies for each lattice size; more precisely, the dataset is biased towards larger lattice sizes. This explains why models $\mathcal{M}_2$ and $\mathcal{M}_3$ perform worse than $\mathcal{M}_1$ for the lattice sizes $L = 5, 6$, since the fraction of measurements for smaller sizes is less represented in their training dataset. However, they perform better for larger grid sizes, where a larger relative number of training tokens are available.

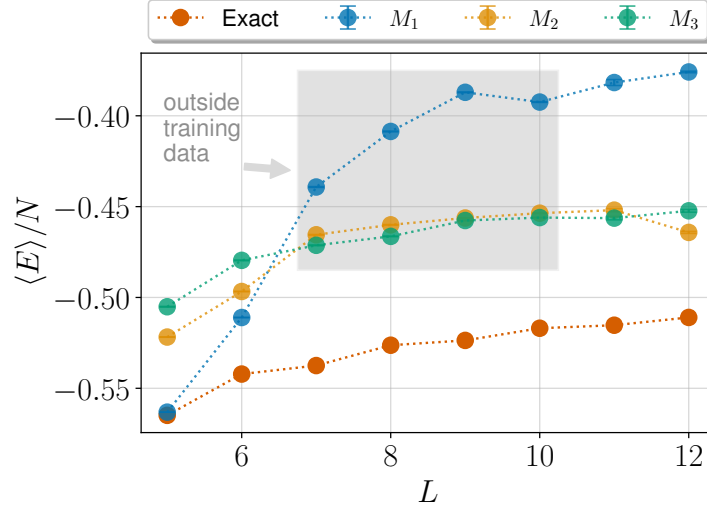


Figure 4.5: **Extrapolation over lattice size L .** The ground-state energy, as estimated by quantum Monte Carlo simulations and the models trained on three different datasets, demonstrates how training data affect the ability to extrapolate to system sizes not included in the training. The diagram is taken from [2] with permission.

4.6 Conclusion

In this chapter, we explored the training of attention-based transformer models for the problem of mapping a system parameters of the Rydberg atom array to the corresponding equilibrium state. We construct an encoder-decoder architecture, where the encoder input is a graph-based representation of the interacting Hamiltonian, and the decoder output represents measurements of the Rydberg occupation.

Since transformers have been developed as autoregressive models for natural language processing, they naturally represent probability distributions, such as Born-rule-distributed measurement outcomes. In the case where a quantum state can be assumed to be pure, real and positive, this probability distribution has a one-to-one mapping to the wavefunction, giving the output of the decoder a powerful interpretation. In the ground state of Rydberg atom arrays governed by Hamiltonian eq. (1.15), this assumption is valid, and we use it to demonstrate that a trained transformer model can accurately reproduce physical estimators in this case.

We have further explored the behavior of the transformer model when the quantum state is no longer real and positive, i.e. in a thermally mixed state at higher temperature. We clearly see the breakdown in our ability to interpret the output of the decoder as the corresponding quantum state, as the physical estimators significantly deviate from their exact values for large T . However, we propose that our decoder architecture could be modified to represent a density matrix [244] or METTS [245, 246], which would be able to represent noisy, thermal and dynamical states, extending the capabilities of the model.

Finally, we attempt to demonstrate the scaling potential of the encoder-decoder transformer. Namely, whether the transformer model, when trained on a dataset composed of smaller many-body system, is able to make predictions for larger system sizes. Our results shows signatures of generalization, however, it also suggest that we have not attained the necessary scale in our transformer to exploit its full potential. Hopefully, by training models with more data on a diverse set of lattice sizes, systematic improvements in model performance will be observed. Indeed, such systematic scaling is the hallmark of large transformers used in industry, giving reason for optimism [247].

More generally, just like in modern LLMs, our transformer architecture should allow for vastly improved performance with increases in the number of trainable parameters λ , larger data sets, and more computational resources. In addition, since these models are being designed to represent physical qubit systems, a number of architectural improvements should be explored. For example, there may be more efficient ways to use tokens than to represent a dictionary of only two individual qubit measurements. In particular, transformers

designed for natural language processing typically manage dictionaries comprising tens to hundreds of thousands of tokens [248]. An alternative approach involves using patches of qubits as tokens, which can reduce some computational demands [179]. However, this method introduces constraints on the system geometries, potentially limiting the applicability to arbitrary quantum array configurations. This balance between computational efficiency and system flexibility remains a critical area for future research.

Notably, all three models discussed in this study were trained using only a single NVIDIA A100 GPU over a duration of 85 hours. Looking forward, we can use this as a benchmark as the size and capability of such transformer models is increased towards the goal of predicting the behavior of larger quantum systems. Inspired by the scaling observed in large language models within industry today, we imagine that significantly larger transformers trained on vastly more data may be able to make highly useful predictions, e.g. about the behavior of a quantum computer in parameter regimes for which no data is currently available.

Finally, while our models in this study learned from synthetic data, our transformer architecture could instead be trained on qubit measurement data obtained from real experimental devices. Such would be the beginning of a new type of *foundation model*, trained on real quantum computer output to generate predictions on a wide variety of unseen input settings. Like today’s LLMs, the capabilities of such large quantum models could increase dramatically with increased number of parameters, data set size, and compute, potentially ushering in a new era of scaling in AI and quantum computer co-design.

Chapter 5

Autoregressive model for continuous variable many-body systems

So far, we have focused on many-body models with discrete degrees of freedoms (DoFs). Applying autoregressive models to such models involves mapping the latent space outputted by the model to a conditional discrete probability distribution with a generator. The generator in this scenario takes the form of a dense linear layer followed by a softmax activation (see section 1.6.3).

For a model that exhibits continuous DoFs, the typical generator would require either a discretization or reparameterization [249] of the space. Instead, let us consider using the flow-based methods introduced in section 1.6.2. These flow-based methods have been exactly developed to handle the parameterization of continuous probability distributions.

In this chapter, we replace the discrete generator with a conditional flow-based generator. Consequently, the model is able to parameterize a joint probability distribution of continuous variables. We apply the model to the classical 2D XY model on a square lattice and study the well known Berezinskii-Kosterlitz-Thouless (BKT) transition [58, 59] that arises in this system. With the proposed modification to the autoregressive models, they are able to accurately capture the properties of the system as it undergoes this phase transition.

5.1 Architecture

We adapt the transformer for continuous probability distributions by replacing the generator with a conditional normalizing flow, shown in fig. 5.1. The flow-based generator

is conditioned on the output of the transformer. This enables the full architecture of the transformer and flow generator to represent joint continuous probability distributions.

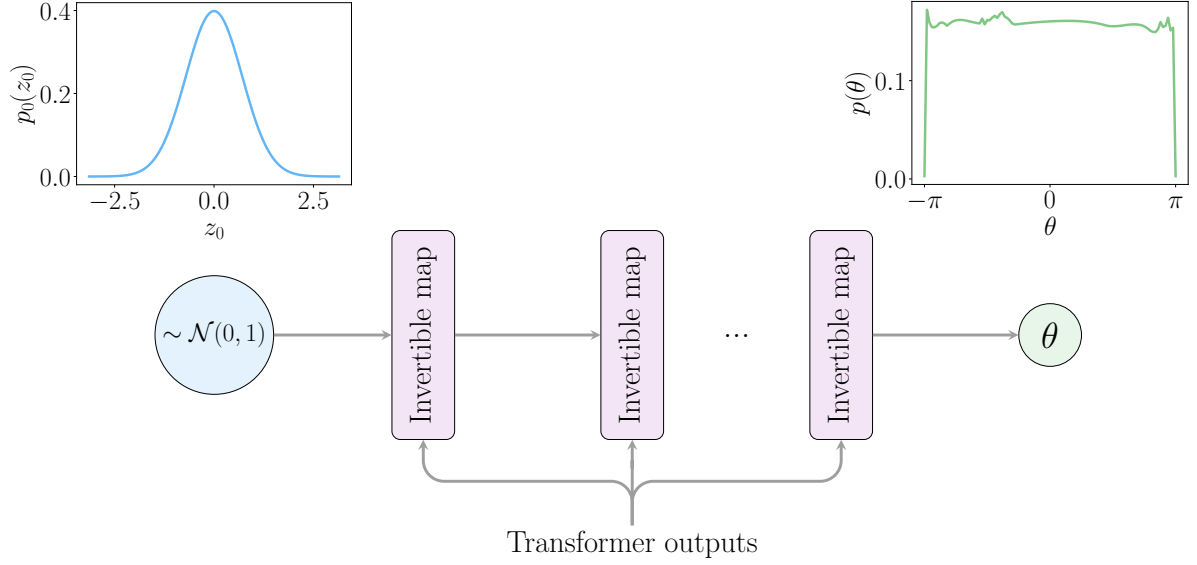


Figure 5.1: **Flow-based generator for the transformer.** The output of the transformer is used to condition all transform layers.

Sampling from a normalizing flow is a simple process consisting of propagating a gaussian sample through the transforms.

$$\theta = f_l \circ f_2 \circ \dots \circ f_{l-1} \circ f_l(z_0), \quad (1.31 \text{ revisited})$$

where, f_i is the i -th transform layer, l is the total number of transform layers. To obtain the probability density of a sample, we first backpropagate the sample in the reverse direction obtaining the initial samples drawn from the standard normal distributions:

$$\begin{aligned} z_{l-1} &= f_l^{-1}(\theta) \\ z_{l-2} &= f_{l-1}^{-1} \circ f_l^{-1}(\theta) \\ &\vdots \\ z_0 &= f_1^{-1} \circ f_2^{-1} \dots f_{l-1}^{-1} \circ f_l^{-1}(\theta). \end{aligned} \quad (5.1)$$

The sample z_0 is drawn from the standard normal distribution, as such, the probability

density is trivial to calculate:

$$p_0(z_0) = \frac{1}{\sqrt{2\pi}} e^{z_0^2}. \quad (5.2)$$

This probability density is then combined with the Jacobians of the parameterized bijective functions to obtain the probability of the transformed sample:

$$\begin{aligned} p(\theta) &= \prod_{i=1}^l \left| \det \left(\frac{df_i}{dz_{i-1}} \bigg|_{z_{i-1}} \right)^{-1} \right| p_0(z_0), \\ \log(p(x)) &= \log(p_0(z_0)) - \sum_{i=1}^l \log \left(\det \left(\frac{df_i}{dz_{i-1}} \bigg|_{z_{i-1}} \right) \right). \end{aligned} \quad (5.3)$$

The models are implemented with PyTorch [192], Lightning [193] and nflows [250]. The hyperparameter tuning was performed with Ray [194, 195].

5.2 Autoregressive model for the classical XY model

The XY model, previously introduced in section 1.1.4, is characterized by the following Hamiltonian:

$$E = \pm \sum_{\langle i,j \rangle} \cos(\theta_i - \theta_j), \quad \theta_i \in [-\pi, \pi). \quad (1.5 \text{ revisited})$$

For a 2D square lattice, the system undergoes the well known BKT transition at $T_c = 0.89273$ [60]. To study the phase transition, we are interested in the energy and helicity modulus. The helicity modulus, also known as the spin stiffness, is an indicator for the BKT transition with the following form:

$$\begin{aligned} \Upsilon_{\alpha} &= \frac{1}{N} \left\langle \sum_{\langle i,j \rangle} \cos(\theta_i - \theta_j) (\alpha \cdot (\mathbf{r}_i - \mathbf{r}_j)) \right\rangle \\ &\quad - \frac{1}{TN} \left\langle \left(\sum_{\langle i,j \rangle} \sin(\theta_i - \theta_j) \alpha \cdot (\mathbf{r}_i - \mathbf{r}_j) \right)^2 \right\rangle, \end{aligned} \quad (5.4)$$

where, $\alpha \in \{\mathbf{x}, \mathbf{y}\}$ is a directional unit vector and $\mathbf{r}_i$ is the position of the i -th rotor. Specifically, the helicity modulus measures the response of the material as a twist is introduced.

The generation of vortices in the model near criticality affects the helicity modulus of the system leading to the phase transition.

We generated data for the 2D XY model on a square lattice with PBCs via MCMC for a variety of sizes and temperatures:

$$L \in \{4, 8, 16\}, \quad T \in [0.1, 2.5]. \quad (5.5)$$

Each dataset consist of 10^4 uncorrelated samples, 80% of which is used for training and the remainder for validation. We train multiple models with parameters outlined in table 5.1 on subsets of the data. We employ the snake path for the autoregressive decomposition of the joint probability distribution based on the discussion in chapter 3.

Transformer with flow generator	
Parameter	Value
N_e	16
N_{heads}	4
N_{FFNN}	256
$N_{\text{blocks}}^{\text{Enc}}$	1
$N_{\text{blocks}}^{\text{Dec}}$	1
Flow transform	Piecewise Rational Quadratic Spline [138, 251]
N_{bins}	32
κ_{tail}	$\pm\pi$
N_{params}	44319

Training	
Parameter	Value
Loss	NLL
Algorithm	Adam
Learning rate	3×10^{-4}
Batch size	256

Table 5.1: **Hyperparameters of the transformer, with flow generator, and the training procedure.** $N_e, N_{\text{heads}}, N_{\text{FFNN}}, N_{\text{blocks}}^{\text{Enc}}$ and $N_{\text{blocks}}^{\text{Dec}}$ are the parameters of the transformer. For the spline-based flow generator, N_{bins} is the number of bins used in the spline, whereas, κ_{tail} is the cutoff that designates the tail region of the distribution.

Instead of the full graph encoder for system parameters, used in the [previous chapter](#), we employ a simplified scheme as the system is governed only by a single global variable, temperature T . We simply feed this single global variable into the transformer encoder and the output is used to condition the transformer decoder.

5.3 Results

We train multiple models for separate sizes of the classical XY model on the 2D square lattice as detailed in [table 4.3](#).

Model	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$
System size N	4×4	8×8	16×16	all
Dataset size D	270000			810000
Epochs	128			256

Table 5.2: **Details on the training dataset of multiple transformer models for different sizes of the XY model.** The dataset size D is the total number of samples, of the system size N , used to train the model for the set of temperatures T in [eq. \(5.5\)](#).

First, we investigate the training of the model with the snake path, the results reinforce the conclusions in [chapter 3](#). The snake path yields a training process that is naturally anisotropic, however, with sufficient training, the model is able to recover the isotropic correlations and converge to the true distribution. For the classical XY model, we observe in [fig. 5.2](#), the helicity modulus reveals a large anisotropy early in training that is resolved with sufficient iterations.

[Figure 5.3](#) demonstrates the models' ability to accurately capture the properties of the dataset. With the simplified conditioning on the system parameter, the model is able to interpolate between values of the temperature T , generalizing to parameter regimes not seen during training. When the model is applied to system sizes not included in the training set, overall trends of the properties for the different system sizes is still consistent. To handle different sizes, we employ two techniques, training on multiple sizes and transfer learning.

We consider training on multiple sizes and transfer learning in order to obtain results for additional sizes. Model $\mathcal{M}_4$ was trained on all the generated data, it is able to accurately reproduce the estimates of the data for all sizes. However, we note that training models

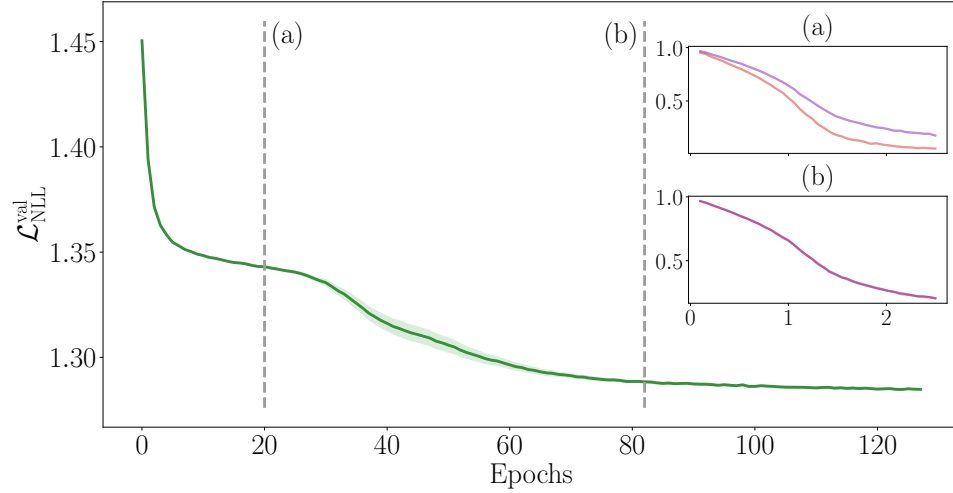


Figure 5.2: **Training behavior of model $\mathcal{M}_2$ on the XY model.** The shaded area corresponds to the standard error of 5 independent training runs. Plots (a) and (b) record the helicity modulus Υ_x and Υ_y , red and purple respectively, against temperature during model training.

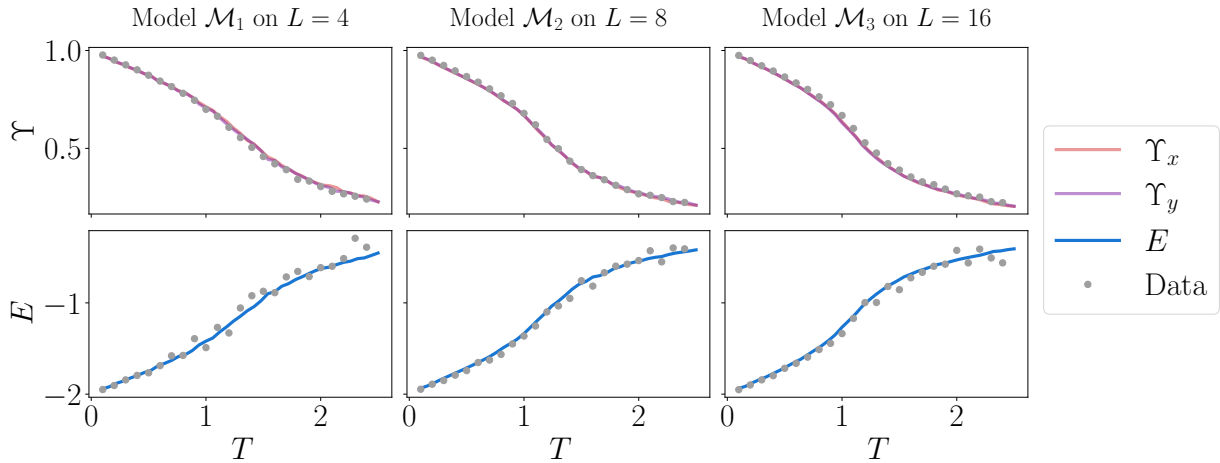


Figure 5.3: **Trained models $\mathcal{M}_n$ estimates for properties of the XY model.** The models are each evaluated on the system sizes they are trained. The estimates for energy and the helicity modulus is obtained from drawing 1000 samples from the model.

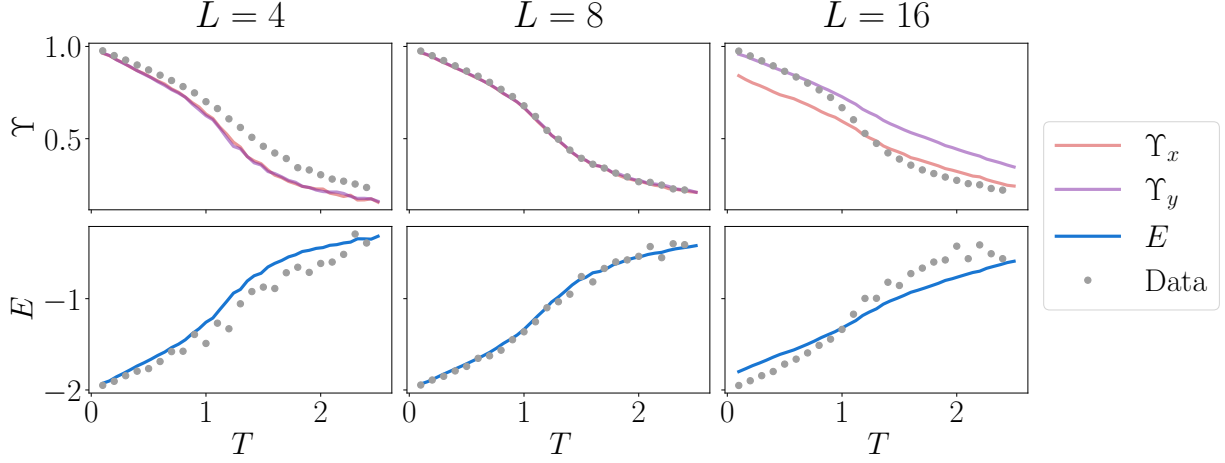


Figure 5.4: **Performance of model $\mathcal{M}_2$ on multiple sizes of the XY model.** The model $\mathcal{M}_2$ is evaluated on system sizes $L = 4, 8, 16$. The estimates for energy and the helicity modulus is obtained from drawing 1000 samples from the model.

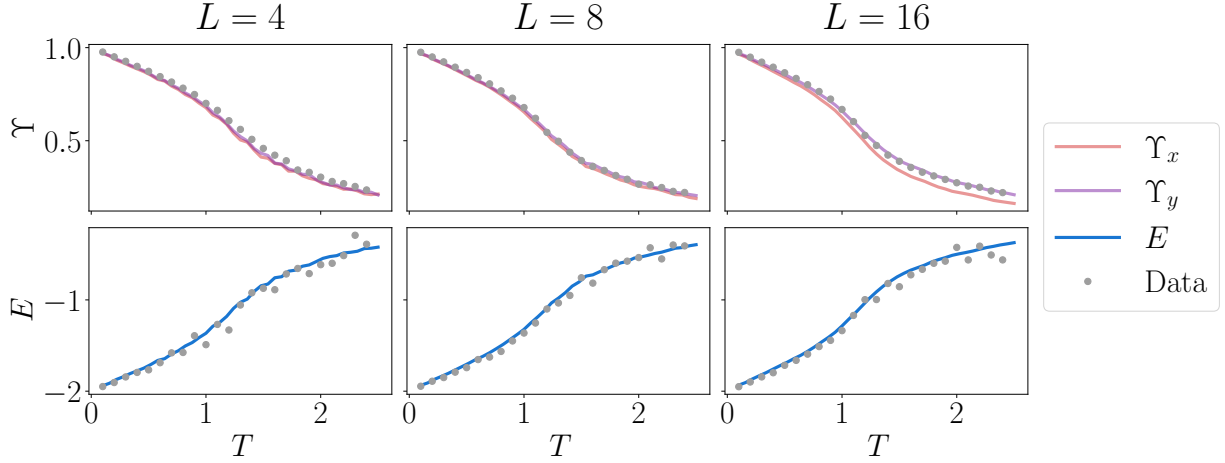


Figure 5.5: **Evaluation of a model $\mathcal{M}_4$ trained on multiple sizes of the XY model.** The model $\mathcal{M}_4$ is evaluated on system sizes $L = 4, 8, 16$. The estimates for energy and the helicity modulus is obtained from drawing 1000 samples from the model.

on multiple sizes require significantly more time. The competition between the losses of varying size constrains the parameter trajectories during the optimization process.

If we instead leverage the fact that the model trained on different sizes has already obtained the qualitative behavior of the properties, we can employ the transfer learning procedure where a pre-trained model is fine tuned for the new systems of interest. We observe that unlike the randomly initialized model, the transfer learned model learns the true distribution at an accelerated rate, most notably, it does not experience the slow down in training due to the need to resolve the anisotropy arising from the choice of path.

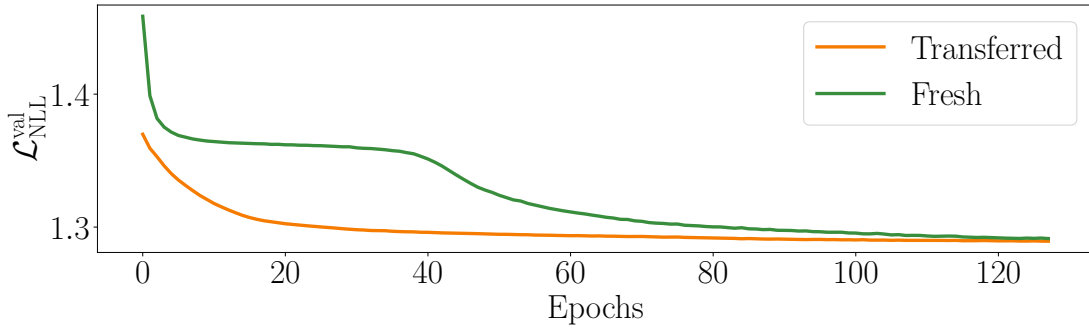


Figure 5.6: **Transfer learning of model $\mathcal{M}_2$ for size $L = 16$ XY model.** The fresh curve corresponds to a model trained from scratch on the size $L = 16$ XY model.

5.4 Conclusion

In this chapter, we introduce a simple modification of typical autoregressive models to be applicable to systems with continuous DoFs. When applied to the system of rotors described by the 2D classical XY model on the square lattice with PBCs, the model was able to accurately capture the behavior of the system throughout the BKT phase transition for the sizes it was trained on. The training results of these models further support the conclusions from chapter 3 regarding the relation between the training behavior and the correlation anisotropy for the snake path in autoregressive models.

However, the models are unable to extrapolate in size, failing significantly when unrolled for larger system sizes. The ability to perform transfer learning in these models helps alleviate some of the cost of training new models for every size. We observe that when transfer learning onto a larger system size, the model is able to bypass the initial phase of training, related to the anisotropic nature of the snake path, converging to the solution earlier.

This work widens the applicability of autoregressive generative models to many-body physics models. There exist formulations of many-body physical models that map systems from discrete space to continuous space, corresponding to statistical field theories. Furthermore, we propose this allows us to study the system in alternate bases, such as the fourier basis. The fourier transform maps local, real-space, elements of the theory to global, momentum space, elements and vice versa. Local constraints of the discreteness of spins does not straightforwardly map to a decomposable momentum space representations. This introduced flexibility, due to the parameterization of continuous DoFs, provides a way to resolve this by considering the full continuous space in which the discrete set of possible momentum representations of the state is contained.

Chapter 6

Fractal transformer: A renormalization group inspired transformer architecture

In the previous chapters, we leverage the ability of sequential models to be unfolded to an arbitrary size to investigate the behavior of the transformer to generalize to larger sizes of the Rydberg system. However, we observed the vanilla transformer struggle to obtain accurate values for the properties of interest.

In this chapter, we take inspiration from renormalization group (RG) [61] and the patched transformer [179] and build an architecture that incorporates the structure of real-space RG. The use of RG in many-body physics has enabled physicist to probe the asymptotic properties of many-body systems, at the thermodynamic limit, by introducing a system for scaling the model that ignores irrelevant components and obtaining a system of equations that describe this scale transformation. In this way, the elements in RG shows promise as a technique for assisting autoregressive models in obtaining generalization in the system size. We show that such a model has a tree-like structure and outperforms the vanilla transformer when extrapolating to a larger size.

6.1 Architecture

In the patched transformer architecture [179], spins are grouped together into patches. The long-range correlations are captured by a transformer, whereas, the short-range correlations

are confined to the patches and are captured by an RNN. The use of the hierarchical structure improves the efficiency of the architecture. As previously mentioned in section 1.6.3, the computational cost of the vanilla transformer scales quadratically in sequence length, in contrast, the RNN computational cost scales linearly with sequence length, as such the reduction in sequence length for the transformer in the patched transformer architecture improves the computational cost. Furthermore, transformer were initially built for natural language processing where the local dimension of the sequence is in excess of 10^3 , whereas, the spin-models of interest to us are only 2 dimensional locally. By patching the spins, the local dimension of the model is increased.

In this work, we generalize the patched transformer for an arbitrary number of hierarchies. Consequently, the resulting architecture resembles a tree. Tree structures have been shown in the field to be beneficial, such as Tree Tensor Networks (TTN) performing better than Matrix Product States (MPS) at reconstructing states [252]. The structure is also similar to the structure imposed by real-space RG shown in fig. 1.4. Instead of individual models at the different hierarchies in the model, we use the same transformer at all levels in the hierarchy and share the weights between them. We call this the fractal transformer.

The fractal transformer is comprised of a vanilla transformer and a FFNN. The FFNN, shown in fig. 6.1, serves as a parameterized decimation scheme, similar to the scale transform in real-space RG. The decimation acts on patch of spins and reduces it to an effective block-spin. Via the iterative application of the decimation to the system, we obtain the coarse-grained spins at various scales:

$$\{\sigma, \sigma', \sigma'', \dots, \sigma^{(\mathcal{O})}\}, \quad (6.1)$$

where, $\mathcal{O}$ is the depth of the fractal transformer. These coarse-grained spins are the inputs to the transformer and the outputs are used as the initial token for the transformer in the subsequent lower level (fig. 6.2). At the leafs of the hierarchy, a generator is applied to map the latent space to a conditional probability distribution. These conditionals combined represent the full joint probability distribution of the spins. The model preserves the autoregressive property of the transformer as each element in the output depends only on the hierarchy of spins that precede it. This corresponds to a pre-order tree traversal where the nodes in the hierarchy are traversed left to right.

Due to the tree-like structure of the fractal transformer, the system sizes N has to be a power of the patch size N_{patch} . Furthermore, the fractal transformer requires that the structure of the lattice is invariant to the patching procedure, this enables further patching of the coarse-grained spins. A compatible lattice would be the hypercubic cubic lattice. For hypercubic lattices with spins on the vertices, a natural choice for the patch is the

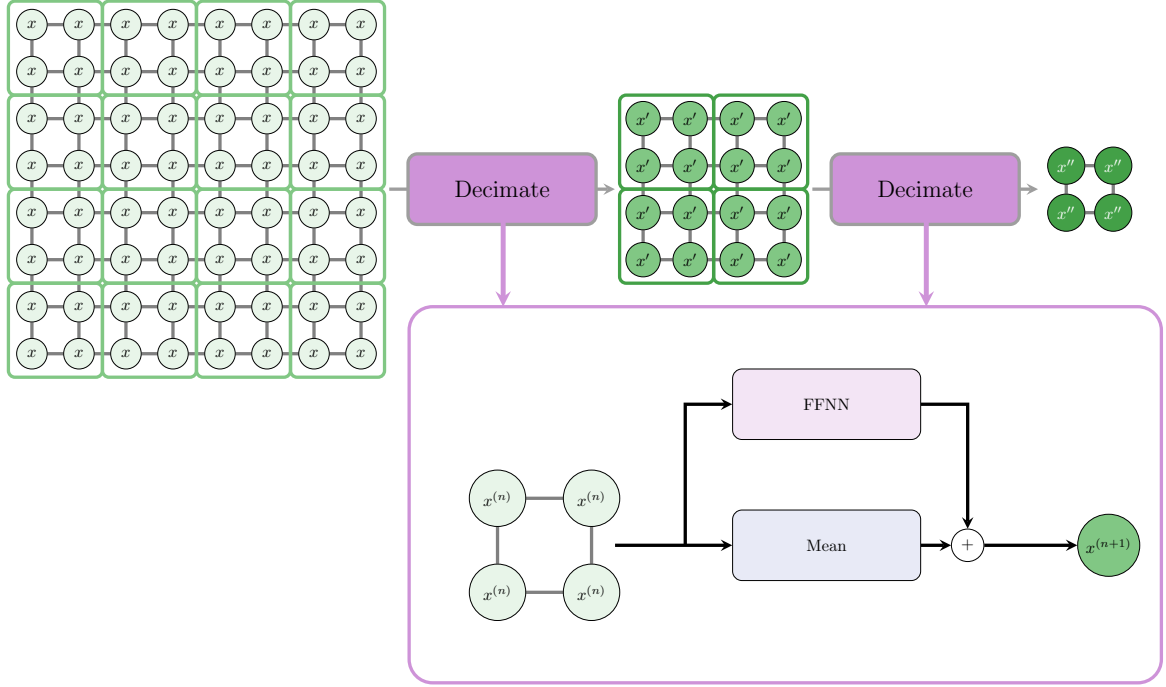


Figure 6.1: Neural network for the decimation of spins in the fractal transformer.

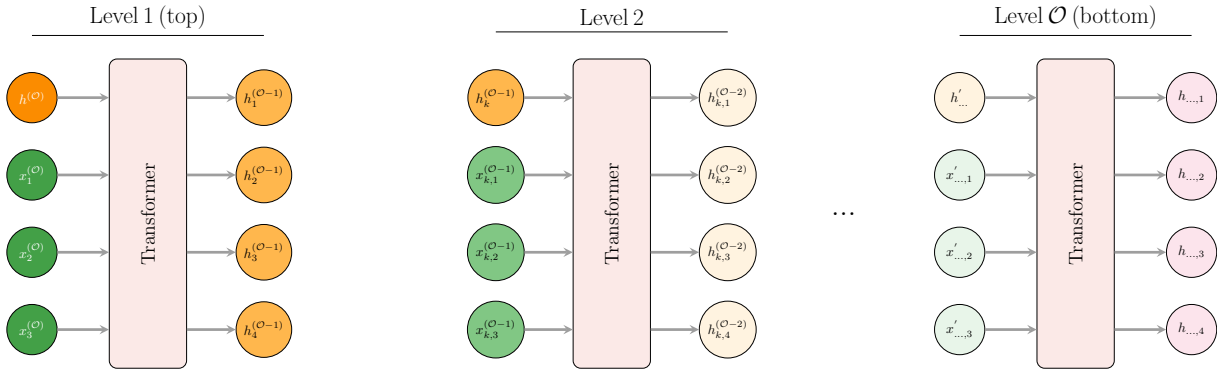


Figure 6.2: Schematic for the hierarchy of transformer calls in the fractal transformer. The top of the hierarchy corresponds with level 1 and takes as input the order $\mathcal{O}$ coarse-grained spins, whereas, level $\mathcal{O}$ consist of leaf nodes at the bottom of the tree with inputs being the actual spins of the system.

hypercubic unit cell. For a d -dimensional hypercubic lattice, the size of the unit cell is 2^d . The hypercubic lattice exhibits self-similarity, i.e. the lattice has the same geometry at all scales. If the spins were to instead live on the edges, faces or hyperfaces, there exist groupings of the spins that result in a hypercubic lattice with the grouped spins on the vertices.

Due to the hierarchical structure of the fractal transformer, the autoregressive path through the system has a hierarchical block structure, We choose to use the generalized snake path and its generalizations for the autoregressive path through the hypercubic patch at all levels in the hierarchy. At the scale of the hypercubic unit cell, the generalized snake path and generalized hilbert path are identical. Therefore, the total path is mostly isotropic.

A key property of sequential models are their ability to handle arbitrary sequence length. This property allows for sequential models to be unrolled, i.e. during inference the model is applied to, or use to generate, longer sequences than present in the training dataset. In our fractal transformer architecture, instead of unrolling in the sequence size, a more natural way to unroll the network is in the hierarchy. This scheme increases the sequence length by a multiplicative factor of the patch size N_{patch} at once instead of by a single element at once.

As the transformer, in the fractal transformer, is constantly called on a fixed size input (the patch size N_{patch}), the overall computational cost of the fractal transformer is govern by the number of calls to the transformer. For an order $\mathcal{O}$ hierarchy, the number of calls is the following:

$$\sum_{i=0}^{\mathcal{O}-1} N_{\text{patch}}^i = 1 + N_{\text{patch}} + N_{\text{patch}}^2 + \dots + N_{\text{patch}}^{\mathcal{O}-1} \propto N, \quad (6.2)$$

Therefore, the fractal transformer is computationally efficient with linear asymptotic scaling with the system size.

Similar to in previous chapters, we are able to modify the transformer to be conditioned on the system parameters with an encoder-decoder architecture. Since the model is conditioned on the system parameters, after training, we are able to evaluate the model on system parameters not present in the training dataset. We use the simple encoding of system parameters introduced in the [previous section](#), the system parameters are directly passed to the transformer encoder.

In the following sections, we test the fractal transformer with hyperparameters described in table [6.1](#) on multiple many body systems.

The models are implemented with PyTorch [\[192\]](#), Lightning [\[193\]](#) and nflows [\[250\]](#). The hyperparameter tuning was performed with Ray [\[194, 195\]](#).

Fractal transformer		Training	
Parameter	Value	Parameter	Value
N_e	16	Loss	NLL
$N_{\text{decimation}}$	8	Algorithm	Adam
N_{heads}	2	Learning rate	1×10^{-3}
N_{FFNN}	128	Batch size	32
N_{blocks}	2	p_{dropout}	0.1

Table 6.1: **Hyperparameters of the fractal transformer architecture and training.**

6.2 Results

6.2.1 Ising model

We train the fractal transformer on the 2D ferromagnetic Ising model on a square lattice described by eq. (1.1) and fig. 1.1 with an MCMC generated dataset. We configure the model with a conditioning on the temperature. We train multiple models with different system sizes, the models are presented in table 6.2.

Model	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$
Order $\mathcal{O}$	2	3	4	3, 4
System size N	4×4	8×8	16×16	$8 \times 8, 16 \times 16$
Dataset size D	500000			500000, 500000
Temperatures T	[0.1, 5]			

Table 6.2: **Details on the dataset used to train multiple fractal transformer models.** The dataset is split 80% for training and 20% for validation and testing.

we study the estimates for the magnetization M , energy E , magnetic susceptibility χ and the specific heat capacity C_v produced by models. The magnetic susceptibility χ and

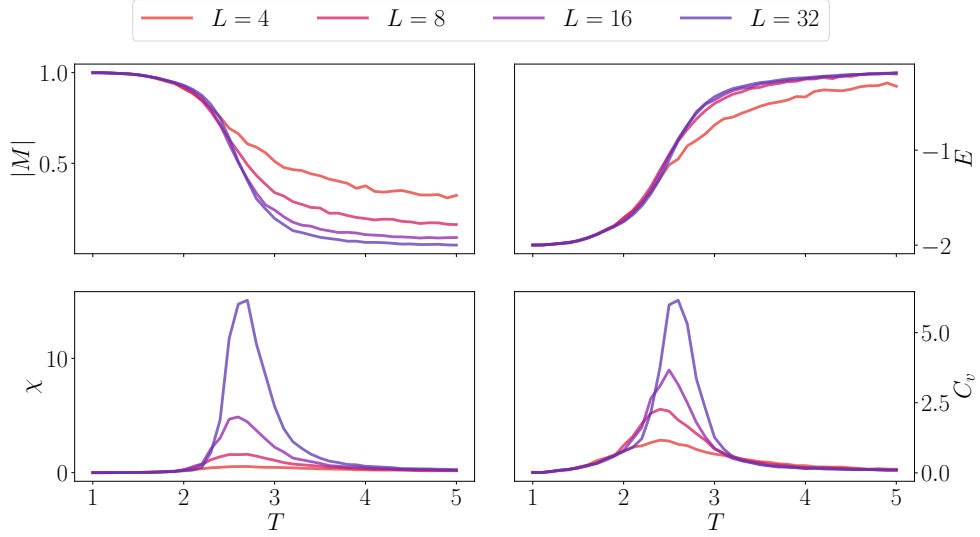


Figure 6.3: **Fractal transformer estimates for the properties of the 2D Ising model across the phase transition.** The model $\mathcal{M}_2$ is evaluated in this figure with 1000 samples per point. The estimated critical temperature is $T_c \approx 2.210(35)$.

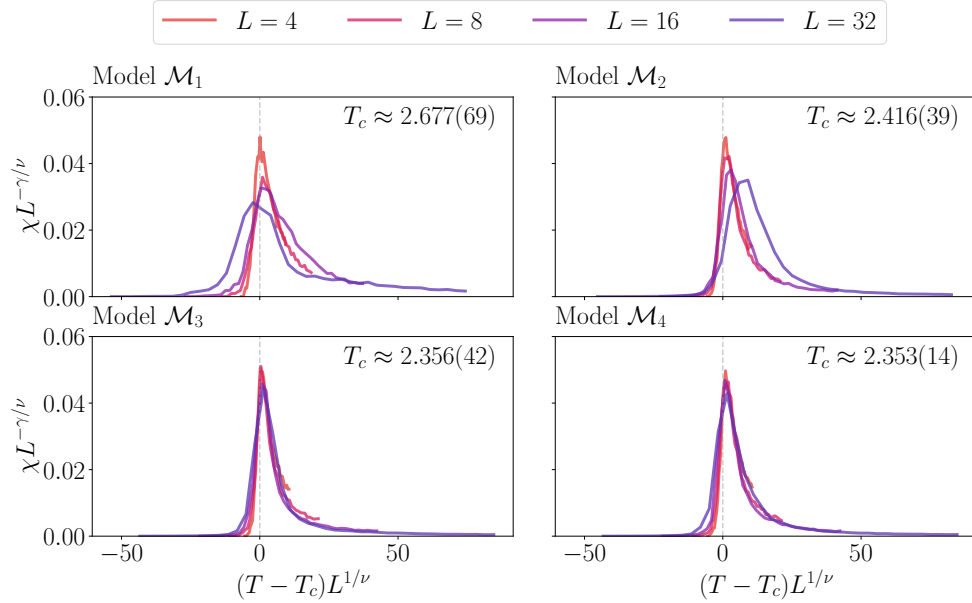


Figure 6.4: **Collapse in the magnetic susceptibility for the 2D Ising model estimated by the fractal transformer.** The models $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3$ and $\mathcal{M}_4$ are evaluated in this figure with 1000 samples per point. Here, we use the well-known exact critical exponents for the 2D Ising model.

specific heat capacity C_v are estimated from the model with the following:

$$\chi = \frac{\langle M^2 \rangle - \langle M \rangle^2}{NT} \quad (6.3)$$

$$C_v = \frac{\langle E^2 \rangle - \langle E \rangle^2}{NT^2} \quad (6.4)$$

During the phase transition, the specific heat capacity C_v diverges at criticality. We obtain an estimate of the critical temperature by determining the peak of the specific heat capacity C_v via optimization based on querying the fractal transformer at various temperatures. The estimated critical temperature for trained models are around $T_c \approx 2.4(1)$. We observed that the architecture has significant error in the estimated critical temperature, this is due to the dataset the model is trained on, from the dataset, the critical temperature is around $T_c \approx 2.3(1)$. Therefore, the model predicted critical temperature agrees with the data and we believe training on a cleaner dataset would result in a corresponding improvement of the model.

Additionally, fig. 6.3 demonstrates that unrolling the model results in reasonable estimates of the properties for values of L larger than the system in the training dataset.

We collapse the magnetic susceptibility χ curve with the theoretical values for the critical exponents in fig. 6.4. As the model is trained on larger sizes, systematic improvement is made. Each model performs well on the subsequent larger system $\mathcal{O} + 1$ with the curves overlapping to a significant degree.

6.2.2 Rydberg system

Next, we look to study the fractal transformer when applied on the Rydberg hamiltonian described by eq. (1.15) and fig. 1.15. We configure the model with a conditioning on the detuning δ/Ω and set $R_b/a = 1.15$, $\Omega = 1$ and $\beta = 64$. We train the model on $N = 16 \times 16$ system, at low temperatures $\beta = 64$. The data is simulated with SSE QMC and we sweep the detuning δ/Ω as per eq. (4.6). The dataset has a total size of 100000 samples, 80% used for training and the rest for validation and testing. We only train the model at low temperatures as, similar to RydbergGPT, the current fractal transformer architecture is incapable of differentiating between mixed states and superposition states, this can be resolved by modifying the model to admit a mixed state representation.

In Figure 6.5, we study the properties of the system, including the density $\langle \hat{n} \rangle$, staggered magnetization $\langle \hat{\sigma}^{stag} \rangle$, x -magnetization $\langle \hat{\sigma}^x \rangle$ and the energy $\langle E \rangle$. There is good agreement

for the staggered magnetization, density and energy between the QMC data and the model. However, we observe a deviation in the x -magnetization.

We are able to extract the model's prediction as system size is scaled. The trend of the energy agrees with expectation, the average energy increases and converge as the system is scaled to larger sizes. Despite this, the finite size effects are not fully captured by the model.

The dataset used includes data-spared regions, the kink in the high detuning regime is within such a region. We are confident the introduction of more training data in the vicinity would remedy the issue.

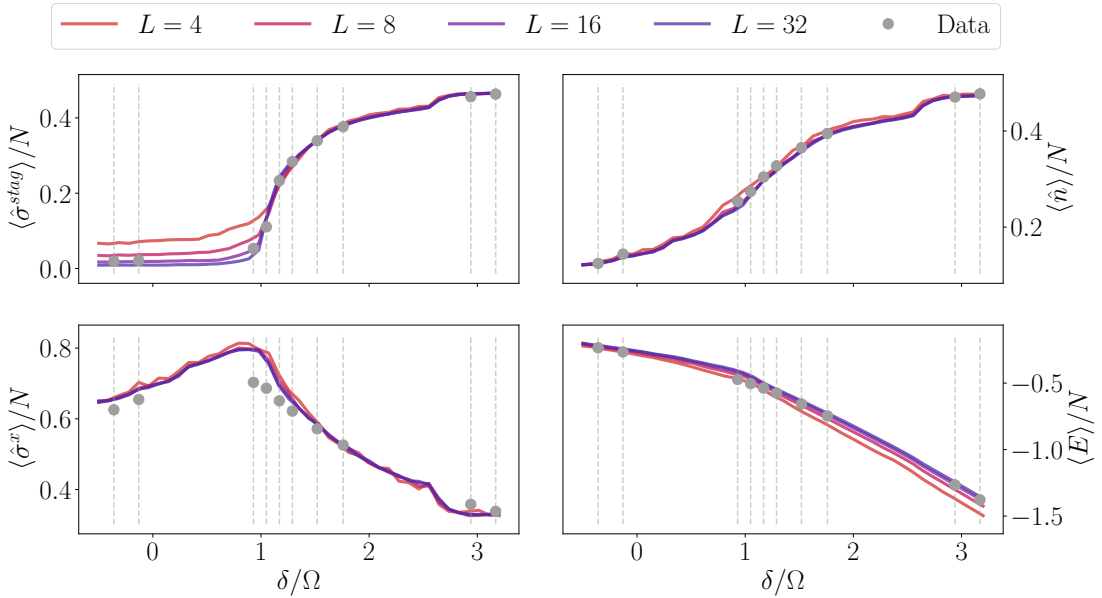


Figure 6.5: **Fractal transformer estimates for the properties of the 2D Ryberg Hamiltonian across the phase transition.** The model is evaluated in this figure with 1000 samples per point. The dashed grey lines indicate the parameters for which the model is trained on.

6.2.3 Classical XY Model

For the XY model described by eq. (1.5), we employ the mechanism introduced in chapter 5, allowing the fractal transformer to parameterize a system of continuous DoFs conditioned

on the temperature of the system. The flow generator is comprised of $N_{\text{layers}} = 2$ layers of piecewise rational quadratic spline transforms [138, 251], with $N_{\text{bin}} = 32$ bins and a tail cutoff at $\kappa_{\text{tail}} = \pm\pi$. We train the model across the BKT transition, with 10^4 uncorrelated samples generated with MCMC for temperatures $T \in [0.1, 2.5]$ and system size $L = 8$.

The continuous fractal transformer was able to achieve similar performance for the helicity modulus and energy at smaller and larger system sizes despite being trained only on a single size $L = 8$. However, there is a slight systematic bias across the different lattice sizes, worse at higher temperatures.

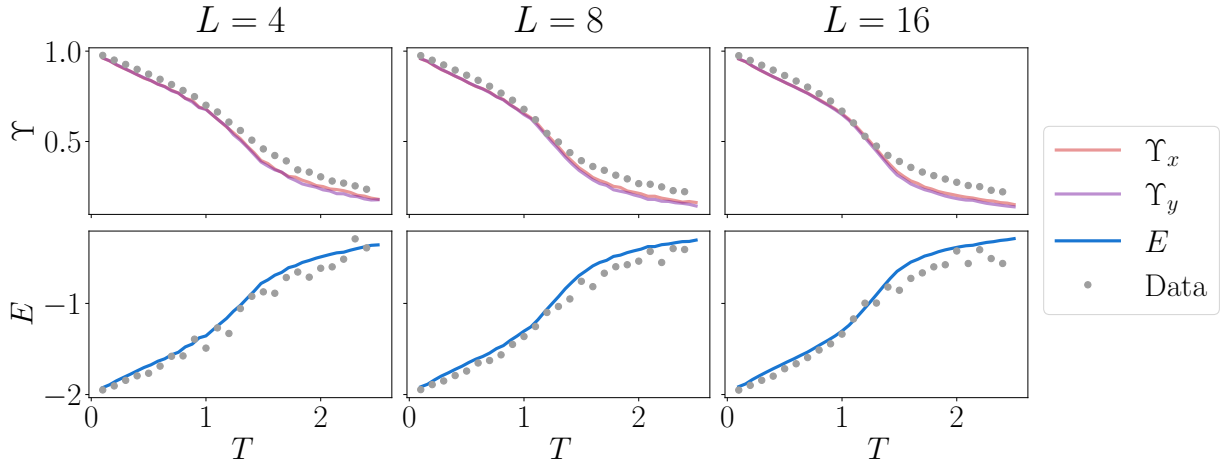


Figure 6.6: **Evaluating the continuous fractal transformer applied to the 2D XY model.** The model is evaluated on system sizes $L = 4, 8, 16$. The estimates for energy and the helicity modulus is obtained from drawing 1000 samples from the model.

6.2.4 Ising lattice gauge theory

The Ising lattice gauge theory, previously introduced in section 1.1.3, is governed by the following hamiltonian:

$$E = \pm \sum_{\square} \sigma_i \sigma_j \sigma_k \sigma_l, \quad \sigma_i \in \{\pm 1\}, \quad (1.4 \text{ revisited})$$

where, the sum is over all possible plaquettes. Figure 6.7 shows the structure of the 3D Ising lattice gauge theory on a cubic lattice with PBCs. The system exhibits a local gauge symmetry, flipping the star conserves energy. By Elitzur's theorem, there does not exist

a local order parameter for the system [57]. The phase transition is characterized by the Wilson loops.

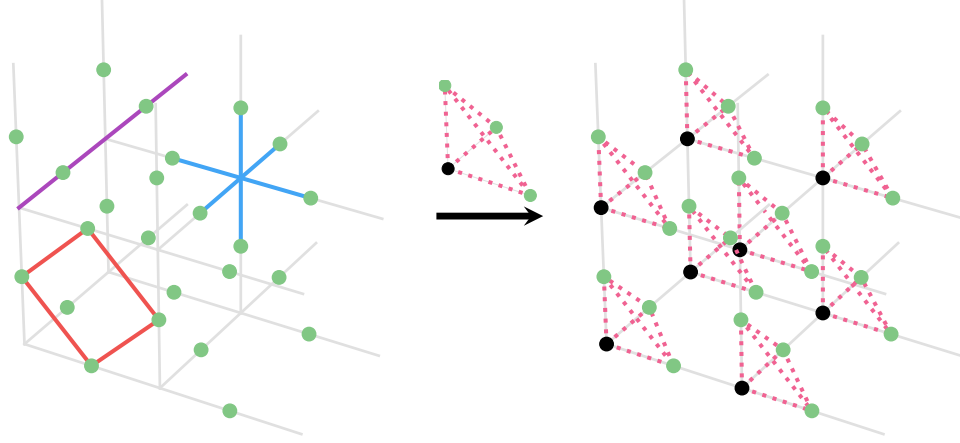


Figure 6.7: **The 3D Ising lattice gauge theory on a cubic lattice and the tetrahedral grouping of spins.** The system has PBCs, on the left, the red square represents a plaquette operator, the blue asterisk a star operator and the purple line a Wilson loop. The spins (green) forming a tetrahedral with the vertices (black) of the lattice are grouped together.

Spins live on the edges in the Ising lattice gauge theory. We obtain a self-similar cubic lattice structure by a tetrahedral grouping of spins around the vertices of the cubic lattice. This reduces the model to a cubic lattice with DoFs on the vertices. The number of possible states for these DoFs on the vertices is $2^3 = 8$. This construction supports the applicability of the fractal transformer on the model.

For the patches of the fractal transformer, we use the cubic unit cell composed of $N_{\text{patch}} = 8$ vertices. The patch is traversed with the generalized 3D snake path. Due to this increase in local dimension, we update the parameters of the fractal transformer model, see table 6.3.

We train the model on MCMC data generated for $L = 4$ for the Ising lattice gauge theory, this corresponds to a total of $N = 3L^3 = 192$ spins, with a conditioning on the temperature. We sweep the temperature $T \in [0.1, 5]$, generating a total to 10^4 uncorrelated samples at each temperature. The model is trained on 80% of the data, while, the remainder is used for validation.

Training the network yields fig. 6.8, we observed that the fractal transformer is unable to generalize the behavior of the Wilson loop to sizes unseen during the training. Furthermore,

Fractal transformer	
Parameter	Value
N_e	128
$N_{\text{decimation}}$	64
N_{heads}	2
N_{FFNN}	1024
N_{blocks}	2
N_{params}	1455113

Table 6.3: **Hyperparameters of the fractal transformer architecture for the 3D Ising lattice gauge theory.**

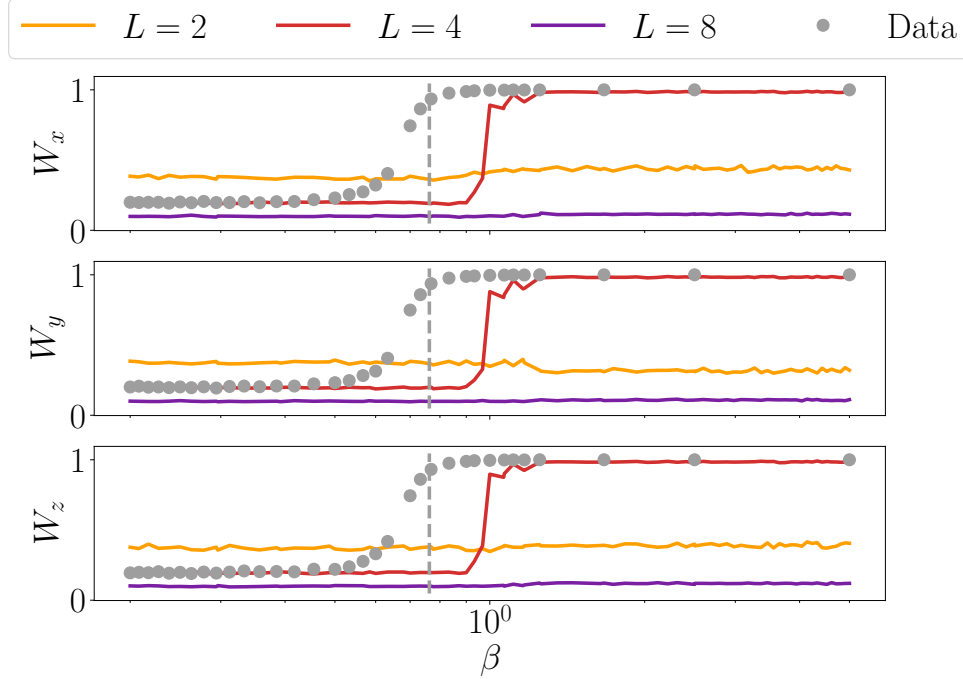


Figure 6.8: **Fractal transformer prediction for the Wilson loops of the Ising lattice gauge theory.** We evaluate the model on multiple sizes $L = 2, 4, 8$. The dashed line indicates the actual critical temperature $\beta_c \approx 0.76$ [56].

for the size the model was trained on, the model struggles to obtain the properties of the system during the phase transition.

Non-local parameters, spanning the full system, cannot be decomposed into smaller blocks. This results in an incompatibility with the hierarchical structure of the fractal transformer, which is obtained by iteratively breaking down the system into smaller and smaller patches. The full size of the system is required to compute the Wilson loop, however, the fractal transformer does not include any information on the full size of the system, just information on the handling of patches and scaling of the system. Furthermore, near criticality, the correlations governing the system are the correlation of all closed contours of spins. These closed contours of spins naturally do not fit into patches and cannot be reduced, as such, the fractal transformer is unable to accurately reproduce states for the system near criticality.

6.3 Conclusion

We demonstrate that a more performant generalization in size can be achieved by the integration of RG structures in autoregressive neural networks. This joins several other works that integrate RG with machine learning [253–257]. The model leverages the RG’s ability to determine the scaling behavior of systems. We test our architecture on a variety of scenarios, including, a system with a quantum phase transition and thermal phase transitions in discrete and continuous systems. Our technique performs well for systems with local order parameters and is not capable of handling systems without local order parameters. This failure is due to the real-space hierarchical structure of the transformer being incompatible with non-local properties of the Ising lattice gauge theory. We project that an alternative formulation of the architecture based instead on momentum-space RG may be able to overcome this.

We have tested the fractal transformer on the 2D square lattice which is self-similar. Future work could expand this with methods to accommodate other lattice geometries, such as triangular, hexagonal, Kagome, etc. The fractal transformer currently consists of a single transformer applied at all levels in the hierarchy. Another avenue for exploration would be to eliminate this weight sharing between levels by modifying the transformer conditioned on the level.

The integration of RG into neural networks is promising and warrants further investigation. Furthermore, in recent years, neural networks have been studied via a mapping to statistical field theories [8–10]. RG has proven a useful tool in the analysis of neural networks [258].

Chapter 7

Conclusion and outlook

In conclusion, We studied the mechanisms and structures underpinning autoregressive generative models by applying them to typical many-body physics models. We explored multiple avenues to achieve efficiency and generality in these machine learning models.

We demonstrated the physicality of the state enforced by neural quantum states (NQSs) allows for a logical tradeoff between computational and sampling complexity. The physicality of the state naturally imparts a constraint on the search space, shrinking it. Whereas, for stateless methods, such as classical shadows and the introduced conditional local density RNN (CLDRNN) architecture, the search space is a superset of the physical states. The incorporation of information is at the heart of the improvement in sampling complexity as it leverages a priori knowledge. With the full Pauli basis set, utilizing this knowledge requires computationally inefficient application of rotations, however, via the further inclusion of information regarding the sign structure of the wavefunction, there exist a regime in which these methods are tractable.

For near-term atom-based quantum processing devices, the repetition rate of experiments limit the amount of extractable data. NQS are a forerunner, facilitating accurate estimates of system properties in spite of finite data. Furthermore, native interactions in these devices include sufficient flexibility to study interesting physical systems amenable to efficient data extraction with NQS.

Optimization of these autoregressive models for many-body physics systems accompanies multiple choices regarding the parameterization. One such parameterization is a product of the sequential property of autoregressive models, requiring the mapping of higher-dimensional systems to 1-dimensional sequences, known as a space-filling path. The choice of this traversal through the higher dimensional discrete space has considerable effects on

the training behavior of such models, implying the existence of an optimal decomposition of the space into a sequence. Contrary to expectations, we find results from autoregressive models that differ from past work examining the same for tensor network methods, suggesting the need for further investigations to reveal the relation between optimal space-filling paths and many-body systems.

Typically, these autoregressive models reconstruct singular states of interest of a many-body system, corresponding to a particular set of system parameters. However, this does not leverage the proven generalization power of these models. Instead, we envision these models learning the equilibrium state for the entirety of the system’s phase diagram and more. This is done via encoding the system parameters as a graph, processing it and, finally, conditioning the autoregressive model on it. This enables us to probe the models’ effectiveness at learning the physics of the many-body system. Our results, applying the method to the Rydberg atom array quantum processor, show promise that with sufficient resources, such models could truly achieve generality.

The usual systems studied by these networks consist of discrete degrees of freedom (DoFs), however, there exist many-body systems that exhibit continuous DoFs. Our approach employ a simple modification of the autoregressive model to handle such systems, augmenting the model output layer with flow-based models, able to capture complex continuous probability distributions. Application of this combination architecture demonstrate the ability to capture the complex behavior of the XY model through the Berezinskii-Kosterlitz-Thouless (BKT) transition.

On the path to achieve scale and generality, we adopt many-body techniques in our model. Specifically, the structure of real-space renormalization group (RG) is introduced into the transformer in an attempt to demonstrate size extrapolation. We observe signatures of the model achieving size independent estimates of the system properties for particular models, with local order parameters. In contrast, the model fails to do the same for models without local order parameters, a famous one being the Ising lattice gauge theory.

The work reveals a host of directions for future research. The Pauli basis form an over-complete basis for the reconstruction pure quantum states, this has been proven achievable with 5 bases [198–200]. Moreover, the bases rotations required are computationally efficient to implement on the NQSs.

Additionally, we have mainly considered global parameters when conditioning autoregressive models on the system parameter graph, there is more to study in regards to local parameters and arbitrary 2-body edge interactions or higher-body hyperedge interactions, corresponding to lattice geometries or more generally graph geometries. Perhaps, the dimension of the system could also be included in the parameterization of the system.

Lastly, in recent years, efforts have been made to integrate RG theory with machine learning [253–258]. We propose introducing the structure of momentum-space RG and other RG formulations into autoregressive models as an area that warrants research.

We are optimistic that the culmination of these areas of research could be revolutionary, yielding novel efficient methods to assist in the understanding of complex many-body systems.

References

- [1] Y. H. Teoh and R. G. Melko, Autoregressive model path dependence near ising criticality (2024), [arXiv:2408.15715 \[cs.LG\]](#) .
- [2] D. Fitzek, Y. H. Teoh, H. P. Fung, G. A. Dagnew, E. Merali, M. S. Moss, B. MacLellan, and R. G. Melko, Rydberggpt (2024), [arXiv:2405.21052 \[quant-ph\]](#) .
- [3] A. Yu. Kitaev, Fault-tolerant quantum computation by anyons, [Annals of Physics](#) **303**, 2 (2003).
- [4] K. Fujii, *Quantum Computation with Topological Codes: From Qubit to Topological Fault-Tolerance*, SpringerBriefs in Mathematical Physics, Vol. 8 (Springer, Singapore, 2015).
- [5] Z.-H. Cui, J. Yang, J. Tölle, H.-Z. Ye, S. Yuan, H. Zhai, G. Park, R. Kim, X. Zhang, L. Lin, T. C. Berkelbach, and G. K.-L. Chan, Ab initio quantum many-body description of superconducting trends in the cuprates, [Nature Communications](#) **16**, 1845 (2025).
- [6] M. Qin, T. Schäfer, S. Andergassen, P. Corboz, and E. Gull, The Hubbard Model: A Computational Perspective, [Annual Review of Condensed Matter Physics](#) **13**, 275 (2022).
- [7] M. T. Schmid, J.-B. Morée, R. Kaneko, Y. Yamaji, and M. Imada, Superconductivity studied by solving ab initio low-energy effective hamiltonians for carrier doped CaCuO_2 , $\text{Bi}_2\text{Sr}_2\text{CuO}_6$, $\text{Bi}_2\text{Sr}_2\text{CaCu}_2\text{O}_8$, and $\text{HgBa}_2\text{CuO}_4$, [Physical Review X](#) **13**, 041036 (2023).
- [8] D. Bachtis, G. Aarts, and B. Lucini, Machine learning with quantum field theories, in *Proceedings of The 38th International Symposium on Lattice Field Theory — PoS(LATTICE2021)*, Vol. 396 (SISSA Medialab, 2022) p. 201.

- [9] J. Halverson, A. Maiti, and K. Stoner, Neural networks and quantum field theory, [Machine Learning: Science and Technology](#) **2**, 035002 (2021).
- [10] M. Demirtas, J. Halverson, A. Maiti, M. D. Schwartz, and K. Stoner, Neural network field theories: Non-Gaussianity, actions, and locality, [Machine Learning: Science and Technology](#) **5**, 015002 (2024).
- [11] S. Ebadi, T. T. Wang, H. Levine, A. Keesling, G. Semeghini, A. Omran, D. Bluvstein, R. Samajdar, H. Pichler, W. W. Ho, S. Choi, S. Sachdev, M. Greiner, V. Vuletić, and M. D. Lukin, Quantum phases of matter on a 256-atom programmable quantum simulator, [Nature](#) **595**, 227 (2021).
- [12] E. Altman, K. R. Brown, G. Carleo, L. D. Carr, E. Demler, C. Chin, B. DeMarco, S. E. Economou, M. A. Eriksson, K.-M. C. Fu, M. Greiner, K. R. Hazzard, R. G. Hulet, A. J. Kollár, B. L. Lev, M. D. Lukin, R. Ma, X. Mi, S. Misra, C. Monroe, K. Murch, Z. Nazario, K.-K. Ni, A. C. Potter, P. Roushan, M. Saffman, M. Schleier-Smith, I. Siddiqi, R. Simmonds, M. Singh, I. Spielman, K. Temme, D. S. Weiss, J. Vučković, V. Vuletić, J. Ye, and M. Zwierlein, Quantum Simulators: Architectures and Opportunities, [PRX Quantum](#) **2**, 017003 (2021).
- [13] C. Monroe, W. C. Campbell, L.-M. Duan, Z.-X. Gong, A. V. Gorshkov, P. W. Hess, R. Islam, K. Kim, N. M. Linke, G. Pagano, P. Richerme, C. Senko, and N. Y. Yao, Programmable quantum simulations of spin systems with trapped ions, [Reviews of Modern Physics](#) **93**, 025001 (2021).
- [14] R. Samajdar, W. W. Ho, H. Pichler, M. D. Lukin, and S. Sachdev, Quantum phases of Rydberg atoms on a kagome lattice, [Proceedings of the National Academy of Sciences](#) **118**, e2015785118 (2021).
- [15] A. Browaeys and T. Lahaye, Many-body physics with individually controlled Rydberg atoms, [Nature Physics](#) **16**, 132 (2020).
- [16] N. Kotibhaskar, C.-Y. Shih, S. Motlakunta, A. Vogliano, L. Hahn, Y.-T. Chen, and R. Islam, Programmable XY-type couplings through parallel spin-dependent forces on the same trapped ion motional modes, [Physical Review Research](#) **6**, 033038 (2024).
- [17] R. Gozalo-Brizuela and E. C. Garrido-Merchan, Chatgpt is not all you need. a state of the art review of large generative ai models (2023), [arXiv:2301.04655 \[cs.LG\]](#) .
- [18] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang, W. Ye, Y. Zhang, Y. Chang, P. S. Yu, Q. Yang, and X. Xie, A Survey on

Evaluation of Large Language Models, [ACM Trans. Intell. Syst. Technol.](#) **15**, 39:1 (2024).

- [19] C. Zhang, C. Zhang, S. Zheng, Y. Qiao, C. Li, M. Zhang, S. K. Dam, C. M. Thwal, Y. L. Tun, L. L. Huy, D. kim, S.-H. Bae, L.-H. Lee, Y. Yang, H. T. Shen, I. S. Kweon, and C. S. Hong, [A Complete Survey on Generative AI \(AIGC\): Is ChatGPT from GPT-4 to GPT-5 All You Need?](#) (2023), [arXiv:2303.11717 \[cs\]](#) .
- [20] OpenAI, [GPT-4 Technical Report](#) (2024), [arXiv:2303.08774 \[cs\]](#) .
- [21] G. Team, [Gemini: A Family of Highly Capable Multimodal Models](#) (2025), [arXiv:2312.11805 \[cs\]](#) .
- [22] DeepSeek-AI, [DeepSeek-V3 Technical Report](#) (2025), [arXiv:2412.19437 \[cs\]](#) .
- [23] J. Carrasquilla and R. G. Melko, Machine learning phases of matter, [Nature Physics](#) **13**, 431 (2017).
- [24] S. Czischek, M. Gärttner, and T. Gasenzer, Quenches near ising quantum criticality as a challenge for artificial neural networks, [Phys. Rev. B](#) **98**, 024311 (2018).
- [25] J. Carrasquilla and G. Torlai, How to use neural networks to investigate quantum many-body physics, [PRX Quantum](#) **2**, 040201 (2021).
- [26] M. Y. Niu, S. Boixo, V. N. Smelyanskiy, and H. Neven, Universal quantum control through deep reinforcement learning, [npj Quantum Information](#) **5**, 1 (2019).
- [27] Y. H. Teoh, M. Drygala, R. G. Melko, and R. Islam, Machine learning design of a trapped-ion quantum spin simulator, [Quantum Science and Technology](#) **5**, 024001 (2020).
- [28] H. Wang, P. Liu, K. Shao, D. Li, J. Gu, D. Z. Pan, Y. Ding, and S. Han, [Transformer-QEC: Quantum Error Correction Code Decoding with Transferable Transformers](#) (2023), [arXiv:2311.16082 \[quant-ph\]](#) .
- [29] J. Bausch, A. W. Senior, F. J. H. Heras, T. Edlich, A. Davies, M. Newman, C. Jones, K. Satzinger, M. Y. Niu, S. Blackwell, G. Holland, D. Kafri, J. Atalaya, C. Gidney, D. Hassabis, S. Boixo, H. Neven, and P. Kohli, Learning high-accuracy error decoding for quantum processors, [Nature](#) **635**, 834 (2024).

- [30] J. Bausch, A. W. Senior, F. J. H. Heras, T. Edlich, A. Davies, M. Newman, C. Jones, K. Satzinger, M. Y. Niu, S. Blackwell, G. Holland, D. Kafri, J. Atalaya, C. Gidney, D. Hassabis, S. Boixo, H. Neven, and P. Kohli, Learning to decode the surface code with a recurrent, transformer-based neural network (2023), [arXiv:2310.05900 \[quant-ph\]](#) .
- [31] K. Albertsson, P. Altoe, D. Anderson, M. Andrews, J. P. Araque Espinosa, A. Aurisano, L. Basara, A. Bevan, W. Bhimji, D. Bonacorsi, P. Calafiura, M. Campanelli, L. Capps, F. Carminati, S. Carrazza, T. Childers, E. Coniavitis, K. Cranmer, C. David, D. Davis, J. Duarte, M. Erdmann, J. Eschle, A. Farbin, M. Feickert, N. F. Castro, C. Fitzpatrick, M. Floris, A. Forti, J. Garra-Tico, J. Gemmler, M. Girone, P. Glaysheer, S. Gleyzer, V. Gligorov, T. Golling, J. Graw, L. Gray, D. Greenwood, T. Hacker, J. Harvey, B. Hegner, L. Heinrich, B. Hooberman, J. Junggeburth, M. Kagan, M. Kane, K. Kanishchev, P. Karpiński, Z. Kassabov, G. Kaul, D. Kcira, T. Keck, A. Klimentov, J. Kowalkowski, L. Kreczko, A. Kurepin, R. Kutschke, V. Kuznetsov, N. Köhler, I. Lakomov, K. Lannon, M. Lassnig, A. Limosani, G. Louppe, A. Mangu, P. Mato, H. Meinhard, D. Menasce, L. Moneta, S. Moortgat, M. Narain, M. Neubauer, H. Newman, H. Pabst, M. Paganini, M. Paulini, G. Perdue, U. Perez, A. Picazio, J. Pivarski, H. Prosper, F. Psihas, A. Radovic, R. Reece, A. Rinkevicius, E. Rodrigues, J. Rorie, D. Rousseau, A. Sauers, S. Schramm, A. Schwartzman, H. Severini, P. Seyfert, F. Siroky, K. Skazytkin, M. Sokoloff, G. Stewart, B. Stienen, I. Stockdale, G. Strong, S. Thais, K. Tomko, E. Upfal, E. Usai, A. Ustyuzhanin, M. Vala, S. Vallecorsa, J. Vassel, M. Verzetti, X. Vilasis-Cardona, J.-R. Vlimant, I. Vukotic, S.-J. Wang, G. Watts, M. Williams, W. Wu, S. Wunsch, and O. Zapata, Machine Learning in High Energy Physics Community White Paper, [Journal of Physics: Conference Series](#) **1085**, 022008 (2018).
- [32] H. Wang, Y. Cao, Z. Huang, Y. Liu, P. Hu, X. Luo, Z. Song, W. Zhao, J. Liu, J. Sun, S. Zhang, L. Wei, Y. Wang, T. Wu, Z.-M. Ma, and Y. Sun, [Recent Advances on Machine Learning for Computational Fluid Dynamics: A Survey](#) (2024), [arXiv:2408.12171 \[cs\]](#) .
- [33] K. Moriwaki, T. Nishimichi, and N. Yoshida, Machine learning for observational cosmology, [Reports on Progress in Physics](#) **86**, 076901 (2023).
- [34] M. Ntampaka, C. Avestruz, S. Boada, J. Caldeira, J. Cisewski-Kehe, R. D. Stefano, C. Dvorkin, A. E. Evrard, A. Farahi, D. Finkbeiner, S. Genel, A. Goodman, A. Goulding, S. Ho, A. Kosowsky, P. L. Plante, F. Lanusse, M. Lochner, R. Mandelbaum, D. Nagai, J. A. Newman, B. Nord, J. E. G. Peek, A. Peel, B. Poczos, M. M.

- Rau, A. Siemiginowska, D. J. Sutherland, H. Trac, and B. Wandelt, [The Role of Machine Learning in the Next Decade of Cosmology](#) (2021), [arXiv:1902.10159 \[astro-ph\]](#).
- [35] E. Ising, Beitrag zur Theorie des Ferromagnetismus, [Zeitschrift für Physik](#) **31**, 253 (1925).
 - [36] R. Peierls, On Ising’s model of ferromagnetism, [Mathematical Proceedings of the Cambridge Philosophical Society](#) **32**, 477 (1936).
 - [37] H. A. Kramers and G. H. Wannier, Statistics of the Two-Dimensional Ferromagnet. Part I, [Physical Review](#) **60**, 252 (1941).
 - [38] L. Onsager, Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition, [Physical Review](#) **65**, 117 (1944).
 - [39] M. Hasenbusch, Finite size scaling study of lattice models in the three-dimensional Ising universality class, [Physical Review B](#) **82**, 174433 (2010).
 - [40] G. S. Pawley, R. H. Swendsen, D. J. Wallace, and K. G. Wilson, Monte Carlo renormalization-group calculations of critical behavior in the simple-cubic Ising model, [Physical Review B](#) **29**, 4030 (1984).
 - [41] K. G. Wilson and M. E. Fisher, Critical Exponents in 3.99 Dimensions, [Physical Review Letters](#) **28**, 240 (1972).
 - [42] A. Pelissetto and E. Vicari, Critical phenomena and renormalization-group theory, [Physics Reports](#) **368**, 549 (2002).
 - [43] L. D. Landau *et al.*, On the theory of phase transitions, *Zh. eksp. teor. Fiz* **7**, 926 (1937).
 - [44] P. C. Hohenberg and A. P. Krekhov, An introduction to the Ginzburg–Landau theory of phase transitions and nonequilibrium patterns, [Physics Reports An Introduction to the Ginzburg–Landau Theory of Phase Transitions and Nonequilibrium Patterns](#), **572**, 1 (2015).
 - [45] R. Puebla, O. Marty, and M. B. Plenio, Quantum Kibble-Zurek physics in long-range transverse-field Ising models, [Physical Review A](#) **100**, 032115 (2019).

- [46] D. Jaschke, K. Maeda, J. D. Whalen, M. L. Wall, and L. D. Carr, Critical phenomena and Kibble–Zurek scaling in the long-range quantum Ising chain, [New Journal of Physics](#) **19**, 033032 (2017).
- [47] M. C. Angelini, G. Parisi, and F. Ricci-Tersenghi, Relations between short-range and long-range Ising models, [Physical Review E](#) **89**, 062120 (2014).
- [48] B.-W. Li, Y.-K. Wu, Q.-X. Mei, R. Yao, W.-Q. Lian, M.-L. Cai, Y. Wang, B.-X. Qi, L. Yao, L. He, Z.-C. Zhou, and L.-M. Duan, Probing Critical Behavior of Long-Range Transverse-Field Ising Model through Quantum Kibble-Zurek Mechanism, [PRX Quantum](#) **4**, 010302 (2023).
- [49] K. Kim, S. Korenblit, R. Islam, E. E. Edwards, M.-S. Chang, C. Noh, H. Carmichael, G.-D. Lin, L.-M. Duan, C. C. Joseph Wang, J. K. Freericks, and C. Monroe, Quantum simulation of the transverse Ising model with trapped ions, [New Journal of Physics](#) **13**, 105003 (2011).
- [50] R. Islam, E. E. Edwards, K. Kim, S. Korenblit, C. Noh, H. Carmichael, G.-D. Lin, L.-M. Duan, C.-C. Joseph Wang, J. K. Freericks, and C. Monroe, Onset of a quantum phase transition with a trapped ion quantum simulator, [Nature Communications](#) **2**, 377 (2011).
- [51] A. Keesling, A. Omran, H. Levine, H. Bernien, H. Pichler, S. Choi, R. Samajdar, S. Schwartz, P. Silvi, S. Sachdev, P. Zoller, M. Endres, M. Greiner, V. Vuletić, and M. D. Lukin, Quantum Kibble–Zurek mechanism and critical dynamics on a programmable Rydberg simulator, [Nature](#) **568**, 207 (2019).
- [52] D.-M. Storch, M. van den Worm, and M. Kastner, Interplay of soundcone and supersonic propagation in lattice models with power law interactions, [New Journal of Physics](#) **17**, 063021 (2015).
- [53] K. G. Wilson, Confinement of quarks, [Physical Review D](#) **10**, 2445 (1974).
- [54] J. B. Kogut, The lattice gauge theory approach to quantum chromodynamics, [Reviews of Modern Physics](#) **55**, 775 (1983).
- [55] F. J. Wegner, Duality in Generalized Ising Models and Phase Transitions without Local Order Parameters, [Journal of Mathematical Physics](#) **12**, 2259 (1971).
- [56] R. Ben-Av, E. Katznelson, S. Solomon, D. Kandel, and P. G. Lauwers, Critical acceleration of Monte-Carlo simulations of Z_2 gauge theory, [Nuclear Physics B - Proceedings Supplements](#) **17**, 285 (1990).

- [57] J. B. Kogut, An introduction to lattice gauge theory and spin systems, [Reviews of Modern Physics](#) **51**, 659 (1979).
- [58] J. M. Kosterlitz and D. J. Thouless, Ordering, metastability and phase transitions in two-dimensional systems, [Journal of Physics C: Solid State Physics](#) **6**, 1181 (1973).
- [59] J. M. Kosterlitz, The critical properties of the two-dimensional xy model, [Journal of Physics C: Solid State Physics](#) **7**, 1046 (1974).
- [60] Y.-D. Hsieh, Y.-J. Kao, and A. W. Sandvik, Finite-size scaling method for the Berezinskii–Kosterlitz–Thouless transition, [Journal of Statistical Mechanics: Theory and Experiment](#) **2013**, P09001 (2013).
- [61] L. P. Kadanoff, Scaling laws for ising models near T_c , [Physics Physique Fizika](#) **2**, 263 (1966).
- [62] K. G. Wilson, Renormalization Group and Critical Phenomena. I. Renormalization Group and the Kadanoff Scaling Picture, [Physical Review B](#) **4**, 3174 (1971).
- [63] K. G. Wilson, Feynman-Graph Expansion for Critical Exponents, [Physical Review Letters](#) **28**, 548 (1972).
- [64] K. G. Wilson, The renormalization group: Critical phenomena and the Kondo problem, [Reviews of Modern Physics](#) **47**, 773 (1975).
- [65] K. G. Wilson and J. Kogut, The renormalization group and the ϵ expansion, [Physics Reports](#) **12**, 75 (1974).
- [66] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, Equation of State Calculations by Fast Computing Machines, [The Journal of Chemical Physics](#) **21**, 1087 (1953).
- [67] W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications, [Biometrika](#) **57**, 97 (1970).
- [68] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Optimization by Simulated Annealing, [Science](#) **220**, 671 (1983).
- [69] E. Marinari and G. Parisi, Simulated Tempering: A New Monte Carlo Scheme, [Europhysics Letters](#) **19**, 451 (1992).

- [70] R. H. Swendsen and J.-S. Wang, Replica Monte Carlo Simulation of Spin-Glasses, [Physical Review Letters](#) **57**, 2607 (1986).
- [71] B. H. Wells, Green's Function Monte Carlo Methods, in [Methods in Computational Chemistry](#) (Springer, Boston, MA, 1987) pp. 311–350.
- [72] A. Sarsa, K. E. Schmidt, and W. R. Magro, A path integral ground state method, [The Journal of Chemical Physics](#) **113**, 1366 (2000).
- [73] C. Cazorla and J. Boronat, Simulation and understanding of atomic and molecular quantum crystals, [Reviews of Modern Physics](#) **89**, 035003 (2017).
- [74] F. Assaad and H. Evertz, World-line and Determinantal Quantum Monte Carlo Methods for Spins, Phonons and Electrons, in [Computational Many-Particle Physics](#), Vol. 739, edited by H. Fehske, R. Schneider, and A. Weiße (Springer Berlin Heidelberg, Berlin, Heidelberg, 2008) pp. 277–356.
- [75] A. W. Sandvik, [Stochastic Series Expansion Methods](#) (2019), [arXiv:1909.10591 \[cond-mat\]](#) .
- [76] R. G. Melko, Stochastic Series Expansion Quantum Monte Carlo, in [Strongly Correlated Systems](#), Vol. 176, edited by A. Avella and F. Mancini (Springer Berlin Heidelberg, Berlin, Heidelberg, 2013) pp. 185–206.
- [77] E. Merali, I. J. S. De Vlucht, and R. G. Melko, Stochastic series expansion quantum Monte Carlo for Rydberg arrays, [SciPost Physics Core](#) **7**, 016 (2024).
- [78] H. F. Trotter, On the Product of Semi-Groups of Operators, [Proceedings of the American Mathematical Society](#) **10**, 545 (1959), 2033649 .
- [79] M. Suzuki, Generalized Trotter's formula and systematic approximants of exponential operators and inner derivations with applications to many-body problems, [Communications in Mathematical Physics](#) **51**, 183 (1976).
- [80] S. L. Sondhi, S. M. Girvin, J. P. Carini, and D. Shahar, Continuous quantum phase transitions, [Reviews of Modern Physics](#) **69**, 315 (1997).
- [81] E. Y. Loh, J. E. Gubernatis, R. T. Scalettar, S. R. White, D. J. Scalapino, and R. L. Sugar, Sign problem in the numerical simulation of many-electron systems, [Physical Review B](#) **41**, 9301 (1990).

- [82] M. Troyer and U.-J. Wiese, Computational Complexity and Fundamental Limitations to Fermionic Quantum Monte Carlo Simulations, [Physical Review Letters](#) **94**, 170201 (2005).
- [83] P. Henelius and A. W. Sandvik, Sign problem in Monte Carlo simulations of frustrated quantum spin systems, [Physical Review B](#) **62**, 1102 (2000).
- [84] M. Takasu, S. Miyashita, and M. Suzuki, Monte Carlo Simulation of Quantum Heisenberg Magnets on the Triangular Lattice, [Progress of Theoretical Physics](#) **75**, 1254 (1986).
- [85] A. D. King, A. Nocera, M. M. Rams, J. Dziarmaga, R. Wiersema, W. Bernoudy, J. Raymond, N. Kaushal, N. Heinsdorf, R. Harris, K. Boothby, F. Altomare, A. J. Berkley, M. Boschnak, K. Chern, H. Christiani, S. Cibere, J. Connor, M. H. Dehn, R. Deshpande, S. Ejtemaee, P. Farré, K. Hamer, E. Hoskinson, S. Huang, M. W. Johnson, S. Kortas, E. Ladizinsky, T. Lai, T. Lanting, R. Li, A. J. R. MacDonald, G. Marsden, C. C. McGeoch, R. Molavi, R. Neufeld, M. Norouzpour, T. Oh, J. Pasvolsky, P. Poitras, G. Poulin-Lamarre, T. Prescott, M. Reis, C. Rich, M. Samani, B. Sheldan, A. Smirnov, E. Sterpka, B. T. Clavera, N. Tsai, M. Volkmann, A. Whitar, J. D. Whittaker, W. Wilkinson, J. Yao, T. J. Yi, A. W. Sandvik, G. Alvarez, R. G. Melko, J. Carrasquilla, M. Franz, and M. H. Amin, [Computational supremacy in quantum simulation](#) (2024), [arXiv:2403.00910 \[quant-ph\]](#) .
- [86] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, Quantum supremacy using a programmable superconducting processor, [Nature](#) **574**, 505 (2019).
- [87] S. Pirandola, B. R. Bardhan, T. Gehring, C. Weedbrook, and S. Lloyd, Advances in photonic quantum sensing, [Nature Photonics](#) **12**, 724 (2018).

- [88] H. Ma, M. Govoni, and G. Galli, Quantum simulations of materials on near-term quantum computers, [npj Computational Materials](#) **6**, 1 (2020).
- [89] R. Ur Rasool, H. F. Ahmad, W. Rafique, A. Qayyum, J. Qadir, and Z. Anwar, Quantum Computing for Healthcare: A Review, [Future Internet](#) **15**, 94 (2023).
- [90] D. P. DiVincenzo, [Topics in quantum computers](#) (1996), [arXiv:cond-mat/9612126 \[cond-mat.mes-hall\]](#) .
- [91] S. Korenblit, D. Kafri, W. C. Campbell, R. Islam, E. E. Edwards, Z.-X. Gong, G.-D. Lin, L.-M. Duan, J. Kim, K. Kim, and C. Monroe, Quantum simulation of spin models on an arbitrary lattice with trapped ions, [New Journal of Physics](#) **14**, 095024 (2012).
- [92] M. Endres, H. Bernien, A. Keesling, H. Levine, E. R. Anschuetz, A. Krajenbrink, C. Senko, V. Vuletic, M. Greiner, and M. D. Lukin, Atom-by-atom assembly of defect-free one-dimensional cold atom arrays, [Science](#) **354**, 1024 (2016).
- [93] D. Barredo, S. de Léséleuc, V. Lienhard, T. Lahaye, and A. Browaeys, An atom-by-atom assembler of defect-free arbitrary two-dimensional atomic arrays, [Science](#) **354**, 1021 (2016).
- [94] F. Rajabi, S. Motlakunta, C.-Y. Shih, N. Kotibhaskar, Q. Quraishi, A. Ajoy, and R. Islam, Dynamical Hamiltonian engineering of 2D rectangular lattices in a one-dimensional ion chain, [npj Quantum Information](#) **5**, 1 (2019).
- [95] W. Paul, Electromagnetic traps for charged and neutral particles, [Reviews of Modern Physics](#) **62**, 531 (1990).
- [96] D. H. E. Dubin, Theory of structural phase transitions in a trapped Coulomb crystal, [Physical Review Letters](#) **71**, 2753 (1993).
- [97] J. P. Schiffer, Phase transitions in anisotropically confined ionic crystals, [Physical Review Letters](#) **70**, 818 (1993).
- [98] B. B. Blinov, D. Leibfried, C. Monroe, and D. J. Wineland, Quantum Computing with Trapped Ion Hyperfine Qubits, [Quantum Information Processing](#) **3**, 45 (2004).
- [99] F. Schmidt-Kaler, S. Gulde, M. Riebe, T. Deuschle, A. Kreuter, G. Lancaster, C. Becher, J. Eschner, H. Häffner, and R. Blatt, The coherence of qubits based on single Ca⁺ ions, [Journal of Physics B: Atomic, Molecular and Optical Physics](#) **36**, 623 (2003).

- [100] T. W. Hänsch and A. L. Schawlow, Cooling of gases by laser radiation, [Optics Communications](#) **13**, 68 (1975).
- [101] C. Monroe, D. M. Meekhof, B. E. King, S. R. Jefferts, W. M. Itano, D. J. Wineland, and P. Gould, Resolved-Sideband Raman Cooling of a Bound Atom to the 3D Zero-Point Energy, [Physical Review Letters](#) **75**, 4011 (1995).
- [102] J. Eschner, G. Morigi, F. Schmidt-Kaler, and R. Blatt, Laser cooling of trapped ions, [JOSA B](#) **20**, 1003 (2003).
- [103] R. Lechner, C. Maier, C. Hempel, P. Jurcevic, B. P. Lanyon, T. Monz, M. Brownnutt, R. Blatt, and C. F. Roos, Electromagnetically-induced-transparency ground-state cooling of long ion strings, [Physical Review A](#) **93**, 053401 (2016).
- [104] P. Schindler, D. Nigg, T. Monz, J. T. Barreiro, E. Martinez, S. X. Wang, S. Quint, M. F. Brandl, V. Nebendahl, C. F. Roos, M. Chwalla, M. Hennrich, and R. Blatt, A quantum information processor with trapped ions, [New Journal of Physics](#) **15**, 123012 (2013).
- [105] F. A. An, A. Ransford, A. Schaffer, L. R. Sletten, J. Gaebler, J. Hostetter, and G. Vittorini, High Fidelity State Preparation and Measurement of Ion Hyperfine Qubits with $|I\rangle = \frac{1}{\sqrt{2}}(|\uparrow\rangle + |\downarrow\rangle)$, [Physical Review Letters](#) **129**, 130501 (2022).
- [106] K. Mølmer and A. Sørensen, Multiparticle Entanglement of Hot Trapped Ions, [Phys. Rev. Lett.](#) **82**, 1835 (1999).
- [107] J. D. Miller, R. A. Cline, and D. J. Heinzen, Far-off-resonance optical trapping of atoms, [Physical Review A](#) **47**, R4567 (1993).
- [108] R. Grimm, M. Weidemüller, and Y. B. Ovchinnikov, Optical Dipole Traps for Neutral Atoms, in *Advances In Atomic, Molecular, and Optical Physics*, Vol. 42, edited by B. Bederson and H. Walther (Academic Press, 2000) pp. 95–170.
- [109] M. Saffman, T. G. Walker, and K. Mølmer, Quantum information with Rydberg atoms, [Reviews of Modern Physics](#) **82**, 2313 (2010).
- [110] D. Jaksch, J. I. Cirac, P. Zoller, S. L. Rolston, R. Côté, and M. D. Lukin, Fast Quantum Gates for Neutral Atoms, [Physical Review Letters](#) **85**, 2208 (2000).
- [111] M. D. Lukin, M. Fleischhauer, R. Cote, L. M. Duan, D. Jaksch, J. I. Cirac, and P. Zoller, Dipole Blockade and Quantum Information Processing in Mesoscopic Atomic Ensembles, [Physical Review Letters](#) **87**, 037901 (2001).

- [112] A. Browaeys, D. Barredo, and T. Lahaye, Experimental investigations of dipole–dipole interactions between a few Rydberg atoms, [Journal of Physics B: Atomic, Molecular and Optical Physics](#) **49**, 152001 (2016).
- [113] X. Wu, X. Liang, Y. Tian, F. Yang, C. Chen, Y.-C. Liu, M. K. Tey, and L. You, A concise review of Rydberg atom based quantum computation and quantum simulation, [Chinese Physics B](#) **30**, 020305 (2021).
- [114] W. S. McCulloch and W. Pitts, A logical calculus of the ideas immanent in nervous activity, [The bulletin of mathematical biophysics](#) **5**, 115 (1943).
- [115] G. Cybenko, Approximation by superpositions of a sigmoidal function, [Mathematics of Control, Signals and Systems](#) **2**, 303 (1989).
- [116] K. Hornik, M. Stinchcombe, and H. White, Multilayer feedforward networks are universal approximators, [Neural Networks](#) **2**, 359 (1989).
- [117] F. Rosenblatt, The perceptron: A probabilistic model for information storage and organization in the brain, [Psychological Review](#) **65**, 386 (1958).
- [118] M.-C. Popescu, V. E. Balas, L. Perescu-Popescu, and N. Mastorakis, Multilayer perceptron and neural networks, [WSEAS Trans. Cir. and Sys.](#) **8**, 579 (2009).
- [119] S. Linnainmaa, Taylor expansion of the accumulated rounding error, [BIT Numerical Mathematics](#) **16**, 146 (1976).
- [120] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, [Automatic differentiation in machine learning: A survey](#) (2018), [arXiv:1502.05767 \[cs\]](#) .
- [121] U. Naumann, Optimal Jacobian accumulation is NP-complete, [Mathematical Programming](#) **112**, 427 (2008).
- [122] R. Courant, Variational methods for the solution of problems of equilibrium and vibrations, [Bulletin of the American Mathematical Society](#) **49**, 1 (1943).
- [123] B. T. Polyak, Some methods of speeding up the convergence of iteration methods, [USSR Computational Mathematics and Mathematical Physics](#) **4**, 1 (1964).
- [124] G. Nakerst, J. Brennan, and M. Haque, [Gradient descent with momentum — to accelerate or to super-accelerate?](#) (2020), [arXiv:2001.06472 \[cs\]](#) .

- [125] T. Tieleman, Lecture 6.5-rmsprop: Divide the Gradient by a Running Average of Its Recent Magnitude (2012).
- [126] J. Duchi, E. Hazan, and Y. Singer, Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, *Journal of Machine Learning Research* **12**, 2121 (2011).
- [127] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization (2014), [arXiv:1412.6980 \[cs.LG\]](#) .
- [128] M. D. Zeiler, [ADADELTA: An Adaptive Learning Rate Method](#) (2012), [arXiv:1212.5701 \[cs\]](#) .
- [129] S. Kullback and R. A. Leibler, On Information and Sufficiency, *The Annals of Mathematical Statistics* **22**, 79 (1951), [2236703](#) .
- [130] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities., [Proceedings of the National Academy of Sciences](#) **79**, 2554 (1982), <https://www.pnas.org/doi/pdf/10.1073/pnas.79.8.2554> .
- [131] P. Tolmachev and J. H. Manton, New Insights on Learning Rules for Hopfield Networks: Memory and Objective Function Minimisation, in [2020 International Joint Conference on Neural Networks \(IJCNN\)](#) (2020) pp. 1–8, [arXiv:2010.01472 \[cs\]](#) .
- [132] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, A learning algorithm for boltzmann machines, [Cognitive Science](#) **9**, 147 (1985).
- [133] M. Á. Carreira-Perpiñán and G. Hinton, On Contrastive Divergence Learning, in *International Workshop on Artificial Intelligence and Statistics* (PMLR, 2005) pp. 33–40.
- [134] S. Geman and D. Geman, Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images, [IEEE Transactions on Pattern Analysis and Machine Intelligence](#) **PAMI-6**, 721 (1984).
- [135] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, Normalizing flows for probabilistic modeling and inference, *J. Mach. Learn. Res.* **22**, 57:2617 (2021).
- [136] I. Kobyzev, S. J. Prince, and M. A. Brubaker, Normalizing Flows: An Introduction and Review of Current Methods, [IEEE Transactions on Pattern Analysis and Machine Intelligence](#) **43**, 3964 (2021).

- [137] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan, [Hybrid Models with Deep and Invertible Features](#) (2019), [arXiv:1902.02767 \[cs\]](#) .
- [138] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, [Neural Spline Flows](#) (2019), [arXiv:1906.04032 \[stat\]](#) .
- [139] T. Müller, B. McWilliams, F. Rousselle, M. Gross, and J. Novák, [Neural Importance Sampling](#) (2019), [arXiv:1808.03856 \[cs\]](#) .
- [140] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, Cubic-Spline Flows, <https://arxiv.org/abs/1906.02145v1> (2019).
- [141] L. Dinh, J. Sohl-Dickstein, and S. Bengio, [Density estimation using Real NVP](#) (2017), [arXiv:1605.08803 \[cs\]](#) .
- [142] A. Sherstinsky, Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network, [Physica D: Nonlinear Phenomena](#) **404**, 132306 (2020).
- [143] R. Pascanu, T. Mikolov, and Y. Bengio, On the difficulty of training recurrent neural networks, in *Proceedings of the 30th International Conference on Machine Learning* (PMLR, 2013) pp. 1310–1318.
- [144] Y. Bengio, P. Simard, and P. Frasconi, Learning long-term dependencies with gradient descent is difficult, [IEEE Transactions on Neural Networks](#) **5**, 157 (1994).
- [145] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, [Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation](#) (2014), [arXiv:1406.1078 \[cs\]](#) .
- [146] S. Hochreiter and J. Schmidhuber, Long Short-Term Memory, [Neural Computation](#) **9**, 1735 (1997).
- [147] H. Shen, Mutual information scaling and expressive power of sequence models (2019), [arXiv:1905.04271 \[cs.LG\]](#) .
- [148] A. Graves, S. Fernandez, and J. Schmidhuber, [Multi-Dimensional Recurrent Neural Networks](#) (2007), [arXiv:0705.2011 \[cs\]](#) .
- [149] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, Attention is all you need, in *Advances in Neural Information Processing Systems*, Vol. 2017-Decem (2017) pp. 5999–6009, [arxiv:1706.03762](#) .

- [150] F. Duman Keles, P. M. Wijewardena, and C. Hegde, On the computational complexity of self-attention, in *Proceedings of The 34th International Conference on Algorithmic Learning Theory*, Proceedings of Machine Learning Research, Vol. 201, edited by S. Agrawal and F. Orabona (2023) pp. 597–619.
- [151] P. J. J. O’Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, A. G. Fowler, E. Jeffrey, E. Lucero, A. Megrant, J. Y. Mutus, M. Neeley, C. Neill, C. Quintana, D. Sank, A. Vainsencher, J. Wenner, T. C. White, P. V. Coveney, P. J. Love, H. Neven, A. Aspuru-Guzik, and J. M. Martinis, Scalable Quantum Simulation of Molecular Energies, *Physical Review X* **6**, 031007 (2016).
- [152] H.-Y. Huang, R. Kueng, and J. Preskill, Predicting many properties of a quantum system from very few measurements, *Nature Physics* **16**, 1050 (2020).
- [153] M. Fannes, B. Nachtergaele, and R. F. Werner, Finitely correlated states on quantum spin chains, *Communications in Mathematical Physics* **144**, 443 (1992).
- [154] S. Rommer and S. Östlund, Class of ansatz wave functions for one-dimensional spin systems and their relation to the density matrix renormalization group, *Physical Review B* **55**, 2164 (1997).
- [155] F. Verstraete, M. M. Wolf, D. Perez-Garcia, and J. I. Cirac, Criticality, the Area Law, and the Computational Power of Projected Entangled Pair States, *Physical Review Letters* **96**, 220601 (2006).
- [156] Y.-Y. Shi, L.-M. Duan, and G. Vidal, Classical simulation of quantum many-body systems with a tree tensor network, *Physical Review A* **74**, 022320 (2006).
- [157] S. R. White, Density-matrix algorithms for quantum renormalization groups, *Physical Review B* **48**, 10345 (1993).
- [158] S. R. White, Density matrix formulation for quantum renormalization groups, *Physical Review Letters* **69**, 2863 (1992).
- [159] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, *Annals of Physics January 2011 Special Issue*, **326**, 96 (2011).
- [160] S. Östlund and S. Rommer, Thermodynamic Limit of Density Matrix Renormalization, *Physical Review Letters* **75**, 3537 (1995).

- [161] J. Hauschild and F. Pollmann, Efficient numerical simulations with Tensor Networks: Tensor Network Python (TeNPy), [SciPost Physics Lecture Notes](#) , 005 (2018).
- [162] D.-L. Deng, X. Li, and S. Das Sarma, Quantum Entanglement in Neural Network States, [Physical Review X](#) **7**, 021021 (2017).
- [163] X.-Q. Sun, T. Nebabu, X. Han, M. O. Flynn, and X.-L. Qi, Entanglement features of random neural network quantum states, [Physical Review B](#) **106**, 115138 (2022).
- [164] D. Iouchtchenko, J. F. Gonthier, A. Perdomo-Ortiz, and R. G. Melko, Neural network enhanced measurement efficiency for molecular groundstates, [Machine Learning: Science and Technology](#) **4**, 015016 (2023).
- [165] C. Hadfield, S. Bravyi, R. Raymond, and A. Mezzacapo, Measurements of Quantum Hamiltonians with Locally-Biased Classical Shadows, [Communications in Mathematical Physics](#) **391**, 951 (2022).
- [166] M. J. S. Beach, I. D. Vlugt, A. Golubeva, P. Huembeli, B. Kulchytskyy, X. Luo, R. G. Melko, E. Merali, and G. Torlai, QuCumber: wavefunction reconstruction with neural networks, [SciPost Phys.](#) **7**, 009 (2019).
- [167] L. L. Viteritti, F. Ferrari, and F. Becca, Accuracy of restricted Boltzmann machines for the one-dimensional J₁-J₂ Heisenberg model, [SciPost Phys.](#) **12**, 166 (2022).
- [168] G. Torlai and R. G. Melko, Latent space purification via neural density operators, [Phys. Rev. Lett.](#) **120**, 240503 (2018).
- [169] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, Neural-network quantum state tomography, [Nature Physics](#) **14**, 447 (2018).
- [170] G. Torlai, B. Timar, E. P. L. van Nieuwenburg, H. Levine, A. Omran, A. Keesling, H. Bernien, M. Greiner, V. Vuletić, M. D. Lukin, R. G. Melko, and M. Endres, Integrating neural networks with a quantum simulator for state reconstruction, [Phys. Rev. Lett.](#) **123**, 230504 (2019).
- [171] X. Gao and L.-M. Duan, Efficient representation of quantum many-body states with deep neural networks, [Nature Communications](#) **8**, 662 (2017).
- [172] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, Reconstructing quantum states with generative models, [Nat. Mach. Intell.](#) **1**, 155 (2019).

- [173] Y. Nomura, A. S. Darmawan, Y. Yamaji, and M. Imada, Restricted boltzmann machine learning for solving strongly correlated quantum systems, [Phys. Rev. B **96**, 205152 \(2017\)](#).
- [174] L. L. Viteritti, R. Rende, and F. Becca, Transformer Variational Wave Functions for Frustrated Quantum Spin Systems, [Phys. Rev. Lett. **130**, 236401 \(2023\)](#).
- [175] L. L. Viteritti, R. Rende, A. Parola, S. Goldt, and F. Becca, Transformer wave function for two dimensional frustrated magnets: Emergence of a spin-liquid phase in the Shastry-Sutherland model, [Physical Review B **111**, 134411 \(2025\)](#).
- [176] M. Hibat-Allah, M. Ganahl, L. E. Hayward, R. G. Melko, and J. Carrasquilla, Recurrent neural network wave functions, [Physical Review Research **2**, 023358 \(2020\)](#).
- [177] M. S. Moss, S. Ebadi, T. T. Wang, G. Semeghini, A. Bohrdt, M. D. Lukin, and R. G. Melko, Enhancing variational monte carlo simulations using a programmable quantum simulator, [Phys. Rev. A **109**, 032410 \(2024\)](#).
- [178] S. Czischek, M. S. Moss, M. Radzihovsky, E. Merali, and R. G. Melko, Data-enhanced variational monte carlo simulations for rydberg atom arrays, [Phys. Rev. B **105**, 205108 \(2022\)](#).
- [179] K. Sprague and S. Czischek, Variational Monte Carlo with large patched transformers, [Communications Physics **7**, 90 \(2024\)](#).
- [180] O. Sharir, Y. Levine, N. Wies, G. Carleo, and A. Shashua, Deep autoregressive models for the efficient variational simulation of many-body quantum systems, [Phys. Rev. Lett. **124**, 020503 \(2020\)](#).
- [181] M. Hibat-Allah, E. M. Inack, R. Wiersema, R. G. Melko, and J. Carrasquilla, Variational neural annealing, [Nature Machine Intelligence **3**, 952–961 \(2021\)](#).
- [182] M. Hibat Allah, Exploring many-body Physics with Recurrent Neural Networks, UWSpace (2023).
- [183] M. Hibat-Allah, R. G. Melko, and J. Carrasquilla, Investigating topological order using recurrent neural networks, [Phys. Rev. B **108**, 075152 \(2023\)](#).
- [184] M. Hibat-Allah, R. G. Melko, and J. Carrasquilla, Supplementing recurrent neural network wave functions with symmetry and annealing to improve accuracy (2024), [arXiv:2207.14314 \[cond-mat.dis-nn\]](#) .

- [185] R. G. Melko and J. Carrasquilla, Language models for quantum simulation, [Nature Computational Science](#) **4**, 11 (2024).
- [186] H. Lange, G. Bornet, G. Emperauger, C. Chen, T. Lahaye, S. Kienle, A. Browaeys, and A. Bohrdt, Transformer neural networks and quantum simulators: A hybrid approach for simulating strongly correlated systems, [Quantum](#) **9**, 1675 (2025), [arXiv:2406.00091 \[cond-mat\]](#) .
- [187] S. Morawetz, I. J. S. De Vlugt, J. Carrasquilla, and R. G. Melko, U(1)-symmetric recurrent neural networks for quantum state reconstruction, [Phys. Rev. A](#) **104**, 012401 (2021).
- [188] H. Wang, M. Weber, J. Izaac, and C. Y.-Y. Lin, Predicting Properties of Quantum Systems with Conditional Generative Models (2022), [arxiv:2211.16943 \[quant-ph\]](#) .
- [189] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, [Science](#) **355**, 602 (2017), <https://www.science.org/doi/pdf/10.1126/science.aag2302> .
- [190] R. G. Melko, G. Carleo, J. Carrasquilla, and J. I. Cirac, Restricted Boltzmann machines in quantum physics, [Nature Physics](#) **15**, 887 (2019).
- [191] A. Seif, K. A. Landsman, N. M. Linke, C. Figgatt, C. Monroe, and M. Hafezi, Machine learning assisted readout of trapped-ion qubits, [Journal of Physics B: Atomic, Molecular and Optical Physics](#) **51**, 174006 (2018).
- [192] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, Pytorch: An imperative style, high-performance deep learning library (2019), [arXiv:1912.01703 \[cs.LG\]](#) .
- [193] W. Falcon and The PyTorch Lightning team, [PyTorch Lightning](#) (2019).
- [194] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica, Tune: A research platform for distributed model selection and training (2018), [arXiv:1807.05118 \[cs.LG\]](#) .
- [195] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, and I. Stoica, [Ray: A Distributed Framework for Emerging AI Applications](#) (2018), [arXiv:1712.05889 \[cs\]](#) .

- [196] G. Torlai, G. Mazzola, G. Carleo, and A. Mezzacapo, Precise measurement of quantum observables with neural-network estimators, [Physical Review Research](#) **2**, 022060 (2020).
- [197] S. Bravyi, D. P. DiVincenzo, R. I. Oliveira, and B. M. Terhal, [The Complexity of Stoquastic Local Hamiltonian Problems](#) (2007), [arXiv:quant-ph/0606140](#) .
- [198] D. Goyeneche, G. Cañas, S. Etcheverry, E. S. Gómez, G. B. Xavier, G. Lima, and A. Delgado, Five Measurement Bases Determine Pure Quantum States on Any Dimension, [Physical Review Letters](#) **115**, 090401 (2015).
- [199] C. Carmeli, T. Heinosaari, M. Kech, J. Schultz, and A. Toigo, Stable pure state quantum tomography from five orthonormal bases, [Europhysics Letters](#) **115**, 30001 (2016).
- [200] L. Zambrano, L. Pereira, and A. Delgado, Improved estimation accuracy of the 5-bases-based tomographic method, [Physical Review A](#) **100**, 022340 (2019).
- [201] M. Bader, [Space-Filling Curves](#), Texts in Computational Science and Engineering, Vol. 9 (Springer, Berlin, Heidelberg, 2013).
- [202] D. Hilbert, Ueber die stetige abbildung einer line auf ein flächenstück, [Mathematische Annalen](#) **38**, 459 (1891).
- [203] R. Niedermeier and P. Sanders, *On the Manhattan-distance between points on space-filling mesh-indexings* (1996).
- [204] G. Morton, [A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing](#) (International Business Machines Company, Ottawa, 1966).
- [205] J. Alber and R. Niedermeier, On multidimensional curves with hilbert property, [Theory of Computing Systems](#) **33**, 295 (2000).
- [206] Y. Bandoh and S. Kamata, An address generator for a 3-dimensional pseudo-Hilbert scan in a cuboid region, in [Proceedings 1999 International Conference on Image Processing \(Cat. 99CH36348\)](#), Vol. 1 (1999) pp. 496–500 vol.1.
- [207] Y. Bandoh and S.-I. Kamata, An address generator, for an N-dimensional pseudo-Hilbert scan in a hyper-rectangular, parallelepiped region, in [Proceedings 2000 International Conference on Image Processing \(Cat. No.00CH37101\)](#), Vol. 1 (2000) pp. 737–740 vol.1, iSSN: 1522-4880.

- [208] S. Kamata and Y. Bandoh, An address generator of a pseudo-Hilbert scan in a rectangle region, in *Proceedings of International Conference on Image Processing*, Vol. 1 (1997) pp. 707–710 vol.1.
- [209] A. E. Nocentino and P. J. Rhodes, Optimizing memory access on gpus using morton order indexing, in *Proceedings of the 48th Annual Southeast Regional Conference*, ACM SE '10 (Association for Computing Machinery, New York, NY, USA, 2010).
- [210] C. Lauterbach, M. Garland, S. Sengupta, D. Luebke, and D. Manocha, Fast bvh construction on gpus, *Computer Graphics Forum* **28**, 375 (2009), <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2009.01377.x>.
- [211] V. Pascucci and R. Frank, Global static indexing for real-time exploration of very large regular grids, in *SC '01: Proceedings of the 2001 ACM/IEEE Conference on Supercomputing* (2001) pp. 45–45.
- [212] D. R. Potapov, Multidimensional data structures usage in adaptive data storages, *Journal of Physics: Conference Series* **1202**, 012020 (2019).
- [213] P. Cha, P. Ginsparg, F. Wu, J. Carrasquilla, P. L. McMahon, and E.-A. Kim, Attention-based quantum tomography, *Machine Learning: Science and Technology* **3**, 01LT01 (2021).
- [214] H. Kim, Y. Zhou, Y. Xu, K. Varma, A. H. Karamlou, I. T. Rosen, J. C. Hoke, C. Wan, J. P. Zhou, W. D. Oliver, Y. D. Lensky, K. Q. Weinberger, and E.-A. Kim, Attention to quantum complexity (2024), [arXiv:2405.11632](https://arxiv.org/abs/2405.11632).
- [215] Y.-H. Zhang and M. Di Ventra, Transformer quantum state: A multipurpose model for quantum many-body problems, *Phys. Rev. B* **107**, 075147 (2023).
- [216] D. Luo, Z. Chen, J. Carrasquilla, and B. K. Clark, Autoregressive neural network for simulating open quantum systems via a probabilistic formulation, *Phys. Rev. Lett.* **128**, 090501 (2022).
- [217] A. Lindenmayer, Mathematical models for cellular interactions in development I. Filaments with one-sided inputs, *Journal of Theoretical Biology* **18**, 280 (1968).
- [218] G. Cataldi, A. Abedi, G. Magnifico, S. Notarnicola, N. D. Pozza, V. Giovannetti, and S. Montangero, Hilbert curve vs Hilbert space: exploiting fractal 2D covering to increase tensor network efficiency, *Quantum* **5**, 556 (2021).

- [219] C. H. Hamilton and A. Rau-Chaplin, Compact hilbert indices: Space-filling curves for domains with unequal side lengths, [Information Processing Letters](#) **105**, 155 (2008).
- [220] Y. Rong, X. Zhang, and J. Lin, Modified hilbert curve for rectangles and cuboids and its application in entropy coding for image and video compression, [Entropy](#) **23**, 10.3390/e23070836 (2021).
- [221] M. Perdacher, C. Plant, and C. Böhm, Improved data locality using morton-order curve on the example of lu decomposition, in [2020 IEEE International Conference on Big Data \(Big Data\)](#) (2020) pp. 351–360.
- [222] L. Li, K. Jamieson, A. Rostamizadeh, E. Gonina, J. Ben-tzur, M. Hardt, B. Recht, and A. Talwalkar, A system for massively parallel hyperparameter tuning, in [Third Conference on Systems and Machine Learning](#) (2020).
- [223] R. Child, S. Gray, A. Radford, and I. Sutskever, Generating long sequences with sparse transformers (2019), [arXiv:1904.10509 \[cs.LG\]](#) .
- [224] N. Kitaev, Łukasz Kaiser, and A. Levskaya, Reformer: The efficient transformer (2020), [arXiv:2001.04451 \[cs.LG\]](#) .
- [225] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, Transformers are RNNs: Fast autoregressive transformers with linear attention, in [Proceedings of the 37th International Conference on Machine Learning](#), Proceedings of Machine Learning Research, Vol. 119, edited by H. D. III and A. Singh (2020) pp. 5156–5165.
- [226] J. Lu, J. Yao, J. Zhang, X. Zhu, H. Xu, W. Gao, C. XU, T. Xiang, and L. Zhang, Soft: Softmax-free transformer with linear complexity, in [Advances in Neural Information Processing Systems](#), Vol. 34, edited by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan (2021) pp. 21297–21309.
- [227] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, Linformer: Self-attention with linear complexity (2020), [arXiv:2006.04768 \[cs.LG\]](#) .
- [228] K. M. Choromanski, V. Likhoshesterov, D. M. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Q. Davis, A. Mohiuddin, L. Kaiser, D. Belanger, L. Colwell, and A. Weller, Rethinking attention with performers, in [accepted to ICLR 2021 \(oral presentation\)](#) (2021).
- [229] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, [Journal of Machine Learning Research](#) **21**, 1 (2020).

- [230] W. Xu, A. V. Venkatramani, S. H. Cantú, T. Šumarac, V. Klüsener, M. D. Lukin, and V. Vuletić, Fast Preparation and Detection of a Rydberg Qubit Using Atomic Ensembles, [Physical Review Letters](#) **127**, 050501 (2021).
- [231] C. Miles, R. Samajdar, S. Ebadi, T. T. Wang, H. Pichler, S. Sachdev, M. D. Lukin, M. Greiner, K. Q. Weinberger, and E.-A. Kim, Machine learning discovery of new phases in programmable quantum simulator snapshots, [Phys. Rev. Res.](#) **5**, 013026 (2023).
- [232] M. Kalinowski, R. Samajdar, R. G. Melko, M. D. Lukin, S. Sachdev, and S. Choi, Bulk and boundary quantum phase transitions in a square rydberg atom array, [Phys. Rev. B](#) **105**, 174417 (2022).
- [233] S. Bravyi, D. P. DiVincenzo, R. I. Oliveira, and B. M. Terhal, The complexity of stoquastic local hamiltonian problems (2007), [arXiv:quant-ph/0606140 \[quant-ph\]](#) .
- [234] R. Samajdar, W. W. Ho, H. Pichler, M. D. Lukin, and S. Sachdev, Complex density wave orders and quantum phase transitions in a model of square-lattice rydberg atom arrays, [Phys. Rev. Lett.](#) **124**, 103601 (2020).
- [235] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, in *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings* (2017) [arxiv:1609.02907](#) .
- [236] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, Dynamic graph cnn for learning on point clouds, [ACM Trans. Graph.](#) **38** (2019).
- [237] J. L. Ba, J. R. Kiros, and G. E. Hinton, Layer normalization (2016), [arXiv:1607.06450 \[stat.ML\]](#) .
- [238] C. Yun, S. Bhojanapalli, A. S. Rawat, S. J. Reddi, and S. Kumar, Are Transformers universal approximators of sequence-to-sequence functions? (2020), [arxiv:1912.10077 \[cs, stat\]](#) .
- [239] M. Fey and J. E. Lenssen, [Fast Graph Representation Learning with PyTorch Geometric](#) (2019), [arXiv:1903.02428 \[cs\]](#) .
- [240] D. Fitzek, Y. H. Teoh, H. P. Fung, G. A. Dagnew, E. Merali, M. S. Moss, B. MacLellan, and R. G. Melko, PennyLane datasets for rydberggpt, <https://pennylane.ai/datasets/other/rydberggpt> (2024).

- [241] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, Vol. 9, edited by Y. W. Teh and M. Titterton (PMLR, Chia Laguna Resort, Sardinia, Italy, 2010) pp. 249–256.
- [242] I. Loshchilov and F. Hutter, Decoupled weight decay regularization, in *International Conference on Learning Representations* (2019).
- [243] I. Loshchilov and F. Hutter, SGDR: Stochastic gradient descent with warm restarts, in *International Conference on Learning Representations* (2017).
- [244] S. Kothe and P. Kirton, Liouville space neural network representation of density matrices (2023), [arXiv:2305.13992 \[quant-ph\]](#) .
- [245] S. R. White, Minimally entangled typical quantum states at finite temperature, *Phys. Rev. Lett.* **102**, 190601 (2009).
- [246] D. Hendry, H. Chen, and A. Feiguin, Neural network representation for minimally entangled typical thermal states, *Phys. Rev. B* **106**, 165111 (2022).
- [247] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, Scaling laws for neural language models (2020), [arXiv:2001.08361 \[cs.LG\]](#) .
- [248] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Nee-lakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, Language models are few-shot learners, in *Advances in Neural Information Processing Systems*, Vol. 33, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Curran Associates, Inc., 2020) pp. 1877–1901.
- [249] L. Wang, Y. Jiang, L. He, and K. Zhou, Continuous-Mixture Autoregressive Networks Learning the Kosterlitz-Thouless Transition, *Chinese Physics Letters* **39**, 120502 (2022).
- [250] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, [nflows: normalizing flows in PyTorch](#) (2020).

- [251] J. A. GREGORY and R. DELBOURGO, Piecewise Rational Quadratic Interpolation to Monotonic Data, [IMA Journal of Numerical Analysis](#) **2**, 123 (1982).
- [252] S. Cheng, L. Wang, T. Xiang, and P. Zhang, Tree tensor networks for generative modeling, [Phys. Rev. B](#) **99**, 155131 (2019).
- [253] X.-Z. Luo, D. Luo, and R. G. Melko, [Operator Learning Renormalization Group](#) (2024), [arXiv:2403.03199](#) [cond-mat, physics:physics, physics:quant-ph] .
- [254] W. Hou and Y.-Z. You, [Machine Learning Renormalization Group for Statistical Physics](#) (2023), [arXiv:2306.11054](#) [cond-mat] .
- [255] S. Efthymiou, M. J. S. Beach, and R. G. Melko, Super-resolving the Ising model with convolutional neural networks, [Physical Review B](#) **99**, 075113 (2019).
- [256] P. Mehta and D. J. Schwab, [An exact mapping between the Variational Renormalization Group and Deep Learning](#) (2014), [arXiv:1410.3831](#) [stat] .
- [257] G. Margalit, O. Lesser, T. Pereg-Barnea, and Y. Oreg, Renormalization-group-inspired neural networks for computing topological invariants, [Physical Review B](#) **105**, 205139 (2022), [arXiv:2202.07669](#) [cond-mat] .
- [258] J. N. Howard, M. Klinger, A. Maiti, and A. G. Stapleton, Bayesian RG flow in neural network field theories, [SciPost Physics Core](#) **8**, 027 (2025).