THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Machine learning for quantum information and computing

Shahnawaz Ahmed

Applied Quantum Physics
Microtechnology and Nanoscience - MC2
Chalmers University of Technology
Göteborg, Sweden 2023

Machine learning for quantum information and computing
Shahnawaz Ahmed

ISBN 978-91-7905-915-6

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 5381
ISSN 0346-718X

Applied Quantum Physics Laboratory
Microtechnology and Nanoscience - MC2
Chalmers University of Technology
SE-412 96 Göteborg, Sweden
Telephone +46 (0)31 772 1000
www.chalmers.se

Author email: shahnawaz.ahmed95@gmail.com

Cover: A generative neural network can find an approximate representation of a quantum state to generate its Wigner function (left), similar to how it can generate artwork (right).

Printed by Chalmers Digitaltryck
Göteborg, Sweden 2023

Machine learning for quantum information and computing
Shahnawaz Ahmed

Applied Quantum Physics Laboratory
Department of Microtechnology and Nanoscience (MC2)
Chalmers University of Technology

Abstract

This compilation thesis explores the merger of machine learning, quantum information, and computing. Inspired by the successes of neural networks and gradient-based learning, the thesis explores how such ideas can be adapted to tackle complex problems that arise during the modeling and control of quantum systems, such as quantum tomography with noisy data or optimizing quantum operations, by incorporating physics-based constraints. We also discuss the Bayesian estimation of a quantum state with uncertainty estimates using physically meaningful priors.

Classical machine learning could inspire new quantum-computing algorithms. One such idea is presented to extend the capabilities of variational quantum algorithms using implicit differentiation, enabling straightforward computation of physically interesting quantities on a quantum computer as a gradient. Implicit differentiation also leads to a novel method to generate multipartite entangled quantum states and allows hyperparameter tuning of quantum machine learning algorithms.

Several new experiments were possible due to the theoretical and numerical techniques developed in the thesis — robust generation of a Gottesman-Kitaev-Preskill and cubic phase state in a 3D cavity, fast process tomography of a new family of superconducting gates with known noise, efficient process tomography of a physical operation implementing a logical gate on a bosonic error-correction code, and the reconstruction of a photoelectron's quantum state.

Keywords: Machine learning, quantum information, quantum computing, quantum machine learning, quantum state tomography, quantum process tomography, generative neural networks, optimization, Bayesian estimation, variational quantum algorithms

Acknowledgements

I would like to extend my sincerest gratitude and appreciation to everyone who played a pivotal role in this journey. First and foremost, I would like to thank my supervisor, Anton Frisk Kockum, whose constant encouragement, unwavering support, and invaluable guidance have been instrumental in my growth. I am truly fortunate to have a supervisor who is not only concerned with my academic growth but also cares deeply about my personal growth. Anton, it has been a pleasure working with you. I could not have asked for a better guide and mentor.

I am also indebted to Göran Johansson for providing me the opportunity and flexibility to explore a loosely defined topic combining two fascinating research fields. A special thanks to Carlos Sánchez Muñoz, who ignited the spark for my first research project, and to Franco Nori, whose encouragement allowed me to explore my ideas freely. I am grateful for my time at RIKEN working with Nathan Shammah, Neill Lambert, and Clemens Gneiting. The experience was transformative in honing my skills and cultivating my passion for quantum physics, machine learning, and scientific computing, ultimately helping me grow as a researcher.

My heartfelt appreciation goes out to my brilliant colleagues at Chalmers. My dearest office-mates Pontus Vikstål and Yu Zheng, along with all others who became close friends along the way — you were a constant source of joy, inspiration, and support throughout my journey. I sincerely thank Ingrid Strandberg, Fernando Quijandría, and Timo Hillmann for their time discussing ideas and for comments on my writing. I am thankful to Marina Kudra, Yong Lu, Christopher Warren, Axel Eriksson, Mikael Kervinen, and Simone Gasparinetti for their help in unraveling experimental data and providing me with a broader perspective beyond theory. My time at the applied quantum physics group at Chalmers has been wonderful both academically and personally.

In life, there are some individuals that you meet along the way that leave a lasting impact. Nathan Shammah is one such person that I am deeply thankful to have met through my scientific journey and become friends with. I am very grateful to my physics professors at BITS, especially Kinjal Banerjee and Radhika Vathsan, for their guidance as an undergraduate. I want to express my sincere gratitude to Gurdeep Singh, who, as an academic mentor, inspired me to pursue a research career. Lastly, my cousin Jay introduced me to the fascinating world of quantum mechanics through the double-slit experiment during high school inspiring me to pursue physics.

I want to thank the exceptional team at Xanadu Quantum Technologies, especially Nathan Killoran and Juan Carrasquilla from UWaterloo and VectorAI. Nathan Killoran's expert insights and suggestions provided invaluable clarity to my research project. I am also grateful for the support, camaraderie, and scientific discussions with the Xanadu Quantum Technologies team over the years.

Additionally, I would like to express my deepest appreciation to my dear mother and father, who have showered me with unwavering love and encouragement. They are the source of my strength and confidence. Their sacrifices have made everything easy for me in my life, inspiring me to follow my dreams. I am truly fortunate to have such remarkable parents who have supported me wholeheartedly and instilled in me the values of resilience and perseverance. The same love and appreciation extends to my family, who have always supported me and pushed me to reach higher.

To my cherished friends, especially from BITS, I am forever grateful for the opportunity to have met you in Goa and grown with you over the years. It was a magical time that influenced every aspect of my life. I have found lifelong friends in you, and your presence has been a driving force inspiring me all along the way. I want to express my wholehearted appreciation to every one of you.

Finally, I would like to give a special thanks to my wife Okka and her family for their support, love, and faith in me. Okka, you have been both the anchor to my ship when I needed stability as well as the sail that allowed me to propel forward towards new adventures.

Göteborg, August 2023
Shahnawaz Ahmed

List of publications

I. **Quantum state tomography with conditional generative adversarial networks**
Shahnawaz Ahmed, Carlos Sánchez Muñoz, Franco Nori, and Anton Frisk Kockum
Physical Review Letters **127**, 140502 (2021). arXiv:2008.03240.

II. **Classification and reconstruction of quantum states with deep neural networks**
Shahnawaz Ahmed, Carlos Sánchez Muñoz, Franco Nori, and Anton Frisk Kockum
Physical Review Research **3**, 033278 (2021). arXiv:2012.02185.

III. **Robust preparation of Wigner-negative states with optimized SNAP-displacement sequences**
Marina Kudra, Mikael Kervinen, Ingrid Strandberg, Shahnawaz Ahmed, Marco Scigliuzzo, Amr Osman, Daniel Pérez Lozano, Mats O. Tholén, Riccardo Borgani, David B. Haviland, Giulia Ferrini, Jonas Bylander, Anton Frisk Kockum, Fernando Quijandría, Per Delsing, and Simone Gasparinetti
PRX Quantum **3**, 030301 (2022). arXiv:2111.07965.

IV. **Implicit differentiation of variational quantum algorithms**
Shahnawaz Ahmed, Nathan Killoran, and Juan Felipe Carrasquilla Álvarez
arXiv:2211.13765.

V. **Pulse-level noisy quantum circuits with QuTiP**
Boxi Li, Shahnawaz Ahmed, Sidhant Saraogi, Neill Lambert, Franco Nori, Alexander Pitchford, and Nathan Shammah
Quantum **6**, 630 (2022). arXiv:2105.09902

VI. **Gradient-descent quantum process tomography by learning Kraus operators**
Shahnawaz Ahmed, Fernando Quijandría, and Anton Frisk Kockum
Physical Review Letters **130**, 150402 (2023). arXiv:2208.00812.

VII. **Extensive characterization and implementation of a family of three-qubit gates at the coherence limit**
Christopher W. Warren, Jorge Fernández-Pendás, Shahnawaz Ahmed, Tahereh Abad, Andreas Bengtsson, Janka Biznárova, Kamanasish Debnath, Xiu Gu, Christian Križan, Amr Osman, Anita Fadavi Roudsari, Per Delsing, Göran Johansson, Anton Frisk Kockum, Giovanna Tancredi, and Jonas Bylander
npj Quantum Information **9**, 44 (2023). arXiv:2207.02938.

VIII. **Quantum process tomography of continuous-variable gates using coherent states**
Mikael Kervinen, Shahnawaz Ahmed, Marina Kudra, Axel Eriksson, Fernando Quijandría, Anton Frisk Kockum, Per Delsing, and Simone Gasparinetti
arXiv:2303.01451.

IX. **Transmon qubit readout fidelity at the threshold for quantum error correction without a quantum-limited amplifier**
Liangyu Chen, Hang-Xi Li, Yong Lu, Christopher W. Warren, Christian J. Križan, Sandoko Kosen, Marcus Rommel, Shahnawaz Ahmed, Amr Osman, Janka Biznárová, Anita Fadavi Roudsari, Benjamin Lienhard, Marco Caputo, Kestutis Grigoras, Leif Grönberg, Joonas Govenius, Anton Frisk Kockum, Per Delsing, Jonas Bylander, and Giovanna Tancredi
npj Quantum Information **9**, 26 (2023). arXiv:2208.05879.

X. **Measuring the quantum state of a photoelectron**
Hugo Laurell, Sizuo Luo, Robin Weissenbilder, Mattias Ammitzböll, Shahnawaz Ahmed, C. L. M. Petersson,Vénus Poulain, Chen Guo, Christoph Dittel, Daniel Finkelstein-Shapiro, Richard J. Squibb, Raimund Feifel, Mathieu Gisselbrecht, Cord L. Arnold, Andreas Buchleitner, Eva Lindroth, Anton Frisk Kockum, Anne L'Huillier, and David Busto
Manuscript in preparation.

Other papers that are outside the scope of this thesis:

A. **PennyLane: Automatic differentiation of hybrid quantum-classical computations**
Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, Shahnawaz Ahmed, Vishnu Ajith, M. Sohaib Alam, Guillermo Alonso-Linaje, B. AkashNarayanan, Ali Asadi, Juan Miguel Arrazola, Utkarsh Azad, Sam Banning, Carsten Blank, Thomas R Bromley, Benjamin A. Cordier, Jack Ceroni, Alain Delgado, Olivia Di Matteo, Amintor Dusko, Tanya Garg, Diego Guala, Anthony Hayes, Ryan Hill, Aroosa Ijaz, Theodor Isacsson, David Ittah, Soran Jahangiri, Prateek Jain, Edward Jiang, Ankit Khandelwal, Korbinian Kottmann, Robert A. Lang, Christina Lee, Thomas Loke, Angus Lowe, Keri McKiernan, Johannes Jakob Meyer, J. A. Montañez-Barrera, Romain Moyard, Zeyue Niu, Lee James O'Riordan, Steven Oud, Ashish Panigrahi, Chae-Yeun Park, Daniel Polatajko, Nicolás Quesada, Chase Roberts, Nahum Sá, Isidor Schoch, Borun Shi, Shuli Shu, Sukin Sim, Arshpreet Singh, Ingrid Strandberg, Jay Soni, Antal Száva, Slimane Thabet, Rodrigo A. Vargas-Hernández, Trevor Vincent, Nicola Vitucci, Maurice Weber, David Wierichs, Roeland Wiersema, Moritz Willmann, Vincent Wong, Shaoming Zhang, and Nathan Killoran
arXiv:1811.04968.

B. **Modelling the ultra-strongly coupled spin-boson model with unphysical modes**
Neill Lambert, Shahnawaz Ahmed, Mauro Cirio, and Franco Nori
Nature Communications **10**, 3721 (2019). arXiv:1903.05892.

C. **BoFiN-HEOM: A bosonic and fermionic numerical hierarchical-equations-of-motion library with applications in light-harvesting, quantum control, and single-molecule electronics**
Neill Lambert, Tarun Raheja, Simon Cross, Paul Menczel, Shahnawaz Ahmed, Alexander Pitchford, Daniel Burgarth, and Franco Nori
Physical Review Research **5**, 013181 (2023). arXiv:2010.10806.

Contents

Preface

"Let there be light!"

The last question
Isaac Asimov

In Isaac Asimov's classic short story "The Last Question", a super-intelligent and self-correcting computer ponders how to save the universe from heat death by reversing entropy. While such a super-intelligent computer does not exist today, machine-learning algorithms, especially using neural networks, are empowering *classical* computers to tackle real-world challenges, from autonomous driving [1], computer vision [2–5], image or text generation [6–11], to even emulating intelligence [12]. Machine learning has also been applied to solve scientific challenges [13–16], discover physical laws [17–19] and new algorithms [20], learning strategies [21], or playing complex games at a superhuman level [22].

However, as classical computing hardware nears atomic scales signaling an end to Moore's law [23], various limitations appear where quantum effects such as tunneling become critical. Furthermore, classical computers struggle with problems such as in quantum chemistry which require simulating physics at the quantum level using classical hardware [24–26]. Therefore new types of computers are being explored — *quantum* computers that harness the power of quantum physics for information processing and computing [27–37].

One might wonder if quantum computers can boost machine learning by providing a computational advantage similar to specialized machine-learning hardware [38–40], as well as whether machine learning can help solve quantum physics problems [41, 42]. This thesis explores these two aspects to set the stage for the results and detailed discussion in the appended papers.

Chapter 1

Introduction

> "Quantum mechanics is certainly imposing. But an inner voice tells me it is not yet the real thing."
>
> Albert Einstein [43]

Quantum mechanics is one of the most successful theories in physics, describing quantum phenomena in systems such as elementary particles [44], atoms [45], electromagnetic fields [46] (light), and even black holes [47]. Yet, its implications bewildered even its creators [43]. Its origins lie in Max Planck's efforts to reconcile experimental data with theory ending up in a radical proposal — energy can only be emitted or absorbed in discrete quanta [48]. In the late 1800s, classical physics predicted that an ideal black body in equilibrium should emit radiation at all frequencies, with more energy radiated as the frequency increases, thereby radiating all its energy instantaneously. The paradox, known as the ultraviolet catastrophe, went against experimental observations and could only be resolved by Planck's counter-intuitive proposal, using which he could derive an equation that fits the data [49].

Quantum mechanics thus emerged through an attempt to fit observed data using a new model for exploring light-matter interaction. Models and fitting data are a central theme in this thesis, as is light. We study the generation, characterization, and transformation of non-classical states of light (e.g., Schrödinger cat states) with machine learning. Let us, therefore, delve into central ideas of quantum physics through light.

1.1 Quantum physics, information, and computing

Light has captivated humanity for ages — from Euclid's optics in around 300 BC to Ibn al-Haytham's Kitab Al Manazir (Book of Optics) introducing the idea of light rays around 1040 AD. Galileo attempted to measure the speed of light at the beginning of the 1600s. In the 1600s, Descartes, Newton, and Huygens proposed conflicting particle and wave theories of light to explain phenomena like reflection and refraction [50]. In the 1800s, Young's experiments supported the wave theory [51], while Faraday speculated that light involved electric and magnetic forces [52]. Maxwell later confirmed this with his theory of electromagnetism [53]. However, in the early 1900s, Planck's idea that energy is emitted and absorbed in discrete units or quanta completely changed our understanding of light and the microscopic world.

Einstein used Planck's ideas to explain the photoelectric effect, suggesting a particle nature of light [54]. Compton's experiments further supported this particle view [55]. These contrasting findings led to wave-particle duality and, eventually, the theory of quantum physics [43]. Quantum physics has since then revolutionized our understanding of nature.

A central idea in quantum physics, beyond quanta and wave-particle duality, is that nature at the quantum level is probabilistic. The state of a quantum system is described by a wavefunction ψ represented by a state vector $|\psi\rangle$ in a complex Hilbert space. Hermitian (self-adjoint) operators $\mathcal{H}$ denote observable quantities such as energy. If $|e_n\rangle$ is an eigenstate of $\mathcal{H}$, we can write

$$\mathcal{H}\,|e_n\rangle = e_n\,|e_n\rangle\,, \tag{1.1}$$

where e_n is an eigenvalue. The complete set of eigenvectors for such a Hermitian operator form an orthonormal basis for the Hilbert space, $\langle e_i|e_j\rangle = \delta_{ij}$, such that any state can be expressed as

$$|\psi\rangle = \sum_n c_n\,|e_n\rangle\,, \tag{1.2}$$

where c_n are complex-valued probability amplitudes. Quantum mechanics postulates that for measurements on such a state, the probability of observing an outcome with eigenvalue e_n is $|c_n|^2$. For example, a measurement operator, such as the photon number operator counts the number of photons in a quantum state.

A two-level quantum system, e.g., a spin or an atom with two energy levels, can be represented with a wave vector using two two basis states

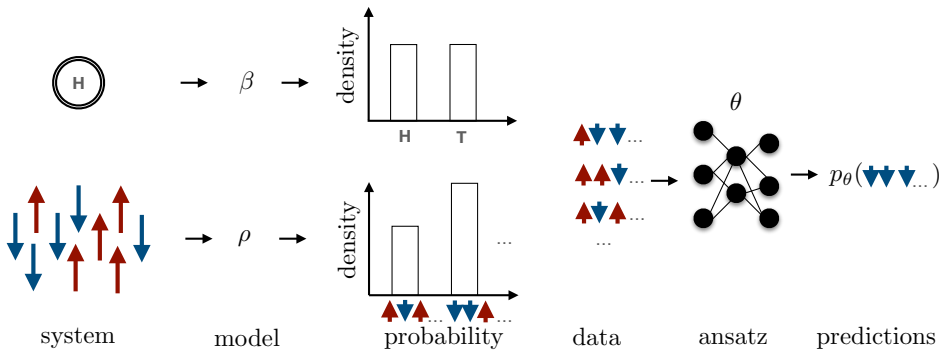


Figure 1.1: A classical coin flip can be modeled as a random experiment with two outcomes (a Bernoulli trial) by specifying the probability of heads. We can observe samples by flipping the coin to infer the probability of heads. Similarly, an interacting quantum system, e.g., spins, can be described by a density matrix ρ that defines probabilities for various measurement outcomes. The dimension of ρ grows exponentially with the system size, making it difficult to tackle larger problems using classical computers, necessitating a tunable quantum device such as a quantum computer. However, if specific constraints exist on the physical system, the probability distribution of outcomes could become simpler. An efficient and flexible ansatz, such as a neural network trained on data, could learn the underlying distribution and predict outcome probabilities or generate samples for outcomes efficiently.

representing the ground ($|0\rangle$) or excited ($|1\rangle$) energy states as

$$|\psi\rangle = c_0\,|0\rangle + c_1\,|1\rangle\,. \tag{1.3}$$

We can also write a quantum state as the linear combination of arbitrary quantum states, e.g., an equal superposition of two states

$$|\psi\rangle = \frac{|\psi_1\rangle \pm |\psi_2\rangle}{\sqrt{2}}, \tag{1.4}$$

where the probability amplitudes can add or get subtracted from each other like amplitudes of a wave, giving rise to the phenomena of superposition and interference. Things get more interesting when we consider multiple such interacting quantum states, see Fig. 1.1. The number of different possible basis states grows exponentially with the number N of such two-level quantum systems. A full classical description of such a state requires keeping track of the $\mathcal{O}(2^N)$ probability amplitudes in general, see Fig. 1.1.

A state where the total wavefunction can be written as a tensor product over different Hilbert spaces is called separable:

$$|\psi\rangle = |\psi\rangle_1 \otimes |\psi\rangle_2 \otimes \dots |\psi\rangle_N \,. \tag{1.5}$$

Generally, we cannot always write a quantum state in the separable form, e.g., the following state for two quantum systems with respective Hilbert spaces H_1 and H_2:

$$|\psi\rangle = \frac{|0\rangle_1 \otimes |0\rangle_2 + |1\rangle_1 \otimes |1\rangle_2}{\sqrt{2}}. \tag{1.6}$$

Such states of a composite quantum system are called entangled as it is impossible to write the state of a subsystem independently from the state of others, i.e., assign a pure state vector to the subsystem. It could be computationally challenging (NP-hard) to even decide whether a quantum state is separable [56, 57]. Various measures of entanglement and separability criteria have been proposed, e.g., a geometric measure of entanglement [58, 59]. In Paper IV, we show how our proposed machine-learning-inspired algorithm can generate entangled quantum states through gradient-based optimization of such a measure of entanglement.

Quantum entanglement was itself a source of great puzzlement [60], showing that quantum systems can have non-trivial correlations that classical systems cannot [61], as confirmed by experiments [62–64]. Separable states, or states with some bounds on a measure of the entanglement, can be written more efficiently, e.g., using tensor networks [65], but general quantum states require tracking all probability amplitudes.

Therefore, the full classical description of quantum systems quickly becomes intractable, and tackling quantum problems with classical computers becomes challenging, although various clever algorithms exist to tackle specific systems [66]. This difficulty prompted the idea of using a tunable quantum system as a quantum computer [27–29]. In recent years, various prototypes of such quantum devices have been demonstrated [33, 67–69] with many interesting applications [36, 37].

There are several approaches to encoding information in quantum systems for computation. A straightforward encoding is simply interpreting $|0\rangle$ and $|1\rangle$ as logical bits and manipulating information by applying a transformation $\mathcal{E}$ as

$$|\psi'\rangle = \mathcal{E}(|\psi\rangle). \tag{1.7}$$

The transformation $\mathcal{E}$ is linear and has certain restrictions to ensure that the coefficients in $|\psi'\rangle$ give valid probabilities. A classical representation of $\mathcal{E}$ can be simply a complex-valued matrix. A quantum computer thus encodes information in the quantum state of a system and manipulates it through linear transformations.

The promise of quantum computers is that we can encode and manipulate information in a superposition of quantum states represented by large state vectors. A quantum operation on this vector can be described by a linear transformation. A quantum computer thus makes it possible to manipulate all the coefficients in a state vector using physical operations.

However, the trouble with quantum physics is that measuring a quantum state only reveals a single bit of information about the state by collapsing it into one of the basis states or their combination with a certain probability. This probabilistic nature only allows us to gather statistics to infer the state from measurement data. Since we are estimating an exponentially large number of coefficients — $\mathcal{O}(d^N)$ for N quantum systems, each with a dimension d — an exponentially increasing number of measurements is necessary.

Therefore, we need to obtain and process an exponentially increasing amount of data to completely characterize quantum systems and operations in a general way such that we can predict the probability of arbitrary measurement outcomes. We approach this problem with tools from machine learning and connect to ideas in generative modeling in the next chapter. But, before that, let us briefly discuss the effect of measurements on a quantum system.

1.2 Measurements, probabilities and informational completeness

Quantum measurement links the quantum description of a state to the macroscopic world. Measurements are performed by letting the quantum system interact with a measuring apparatus that gives a classical outcome. The von Neumann model [70] assumes that the measurements are stochastic, representing the projection of the quantum state into an eigenstate. Multiple repetitions of the measurement on identically prepared copies are necessary to gather enough statistics to infer a quantum state.

Measurements are described using operators and the probability distribution of the measurement outcomes can be computed using a so-called

density operator ρ. The density operator is a matrix that describes a statistical mixture of pure quantum states:

$$\rho = \sum_i p_i \left|\psi_i\right\rangle\!\left\langle\psi_i\right| , \tag{1.8}$$

where $\sum p_i$ denotes the statistical probability for each possible state, different from the intrinsic probability amplitudes of a pure state vector. If we expand the state vectors using the basis $\{|e_n\rangle\}$, the density matrix is characterized by the complex-valued coefficients ρ_{mn} given by

$$\rho = \sum \rho_{mn} \left|e_m\right\rangle\!\left\langle e_n\right| . \tag{1.9}$$

A projection operator $P_n = |e_n\rangle\langle e_n|$ such as $|0\rangle\langle 0|$ extracts the probability for the measurement outcome e_n as

$$p(e_n) = \mathrm{tr}[P_n \rho]. \tag{1.10}$$

Positive-operator-valued-measures (POVMs) generalize measurements with a set of operators $\{\mathcal{M}_n = P_n^\dagger P_n\}$ satisfying $\sum_n \mathcal{M}_n = \mathbb{I}$.

A general observable with eigenvalues a_i and eigenvectors $|a_i\rangle$ can be written as $\mathcal{O} = \sum_i a_i \left|a_i\right\rangle \left\langle a_i\right|$ relying on the Hermiticity of physical observables. The expectation value of the measurement $\mathcal{O}$ is given by the so-called Born rule

$$\begin{aligned} \langle \mathcal{O} \rangle &= \sum_i a_i p(a_i) \\ &= \sum_i a_i \,\mathrm{tr}[\left|a_i\right\rangle \left\langle a_i\right| \rho] \\ &= \mathrm{tr}\left[\sum_i a_i \left|a_i\right\rangle \left\langle a_i\right| \rho\right] \\ &= \mathrm{tr}[\mathcal{O}\rho]. \end{aligned} \tag{1.11}$$

Therefore quantum measurements provide us a way to compute expectation values for measurements. In an experiment, we obtain outcomes distributed according to the Born rule. The task of estimating a classical description of the state, i.e., determining the full density matrix ρ, is called quantum state tomography.

Quantum state tomography is difficult due to the large number of measurements that must be carried out experimentally. However, there is also

a computational difficulty where data processing could take hours or even weeks [71, 72]. The problem is even more pronounced in quantum process tomography, where we wish to estimate how quantum states transform [73]. We tackle such problems in Papers I, II, VI, VII, VIII, and X, exploring efficient reconstruction algorithms for quantum tomography. However, to understand the source of difficulty in estimating quantum systems, we need to discuss the so-called informational completeness of measurements.

Informational completeness (IC) for a set of measurements means that we can uniquely and unambiguously determine a quantum state using this set. Since repeating measurements require several copies of the state, we can think about the sample complexity for tomography in terms of the number of copies of a quantum state necessary for estimation. However, choosing the measurement setting is essential and not trivial and it could depend on many factors, e.g., the assumption of low rank [74–77]. Machine learning has been applied to tackle challenges in certifying IC from raw data [78].

If we ignore the task of determining the measurements that guarantee IC and only focus on the number of copies of ρ, we can do a resource estimation for quantum state tomography. The upper bound for the number of copies required for the tomography of a $d-$dimensional mixed quantum state ρ was known to be $O(d^4)$ [79, 80]. Ref. [81] improved it to $O(d^3)$, but only in Ref. [82] was it shown that the optimal number of copies for full quantum state tomography was $O(d/\epsilon)$ for the estimate ρ' to be within an error ϵ in the Frobenius norm $||\rho' - \rho||_F^2$. In the trace distance, $\mathcal{O}(\text{rank}(\rho)d/\epsilon^2)$ copies are required to obtain an ϵ-accurate estimate.

IC measurements given by a POVM $\{\mathcal{O}_i\}$ allow the computation of the expectation value of any arbitrary observable by completely specifying ρ. Such an IC-POVM consists of at least d operators $\{\mathcal{O}_i\}$ that span the full Hilbert space of dimension d [83, 84]. For example, working in the Fock basis to write the quantum state of light, we often set a cutoff N_c on the Hilbert-space dimension. A POVM measurement can reveal the occupation probability for different Fock basis elements $|n\rangle$, thereby fixing the $O(N_c^2)$ real values needed for reconstructing ρ. Even for moderate cutoffs, such problems can become challenging, requiring many measurements for IC.

The design of IC POVMs is also not straightforward for arbitrary quantum states and higher dimensions. The class of symmetric, informationally complete POVMs (SIC-POVMs) denotes the optimal IC-POVMs with exactly d^2 elements [85]. There are exactly determined SIC-POVMs found by

hand and computer algebra systems using supercomputers for dimensions as high as $d = 844$ [86]. For the case of optical quantum states, Ref. [83] describes a numerical procedure to determine the optimal measurement settings. A geometric interpretation is presented that can guide the choice of the measurement setting for a Schrödinger-cat state. Nevertheless, it is not straightforward to determine the best possible SIC-POVMs to reconstruct a quantum state completely, and often we might use informationally over-complete measurements [87]. Informationally over-complete measurements are ICs that could have more than d^2 outcomes, e.g., measuring a set of qubits locally in all combinations of Pauli operators $(\mathbb{I} \pm \sigma_{x,y,z})/6$.

It is possible that certain quantum states allow efficient descriptions where it is unnecessary to keep track of all the coefficients, e.g., separable quantum states, or coherent states of light which can be described by only specifying the mean photon number or superpositions of coherent states called Schrödinger cat states. Similarly, many other states with rotational symmetry [88, 89] admit an efficient description with a few parameters, see Fig. 1.2.

Further assumptions such as having a low rank of the density matrix, i.e., the density matrix is a mixture of only a small number of pure states, could simplify estimation by reducing the number of independent parameters. Machine-learning methods could therefore be effective in solving learning problems in quantum physics by exploiting patterns in the data [42].

1.3 Machine learning meets quantum physics

Machine-learning algorithms are designed such that they can automatically learn and improve their performance on a task with experience. This is usually achieved by training the algorithm with data, allowing it to recognize and exploit patterns in the data. With the availability of more data, new machine-learning algorithms running on improved hardware such as graphics processing units (GPUs) are solving problems that posed significant issues for computers before.

Face recognition, automated driving, and natural language processing are challenging to solve using hand-crafted algorithms. However, machine-learning algorithms can tackle such tasks [1, 91–93], even achieving super-human performance [94, 95], and showing some level of creativity [12, 96, 97].

The backbone of many such machine-learning algorithms today are

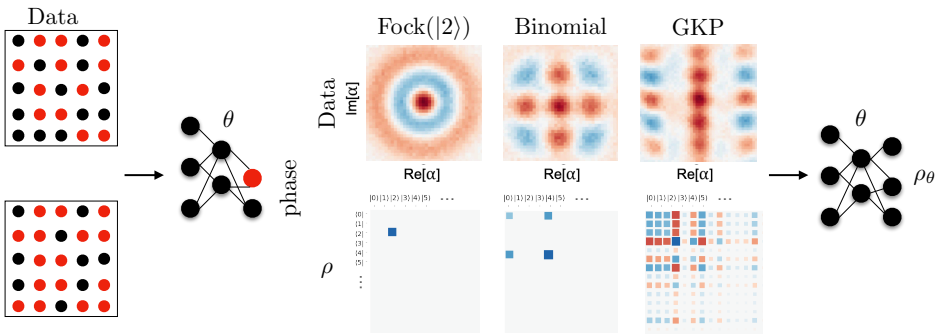


Figure 1.2: Machine learning can help solve problems such as the classification of quantum phases [90], where patterns in the data can be learned by a model, e.g., sampled spin configurations. The model could be a neural network with tunable parameters θ trained to predict a class for a given spin configuration. In the case of non-classical states of light, similar patterns exist in the Wigner function that can be measured experimentally, e.g., pure single-photon Fock states have Wigner functions with rotational symmetry consisting of rings. The density matrix for such a Fock state will have only one non-zero element. A machine-learning algorithm can therefore learn to predict the probability measurement outcomes by approximating or estimating the density matrix.

neural networks that are loosely inspired by the structure of human brains and seemingly emulate intelligence [12]. Beyond solving tasks that could be easy for humans, such as driving a car, neural-network-based algorithms have also been applied to defeat or match humans at various types of games [94, 98]. Self-learning algorithms can even discover the rules of games by themselves [95]. More recently, machine-learning techniques have also been applied successfully to solve grand scientific challenges such as the protein-folding problem, outperforming other human-developed methods and models [99].

Therefore, combining quantum physics and machine learning can lead to interesting new ideas and applications. From a quantum description of light and its characterization, we can now discuss how machine learning can benefit quantum physics. Recent studies have demonstrated the promising potential of machine-learning-inspired techniques for addressing challenges in quantum information processing [42]. These techniques have shown success in tasks such as faster automated calibration and control [100], quantum experiment design [101], and reducing error rates in the readout of information from quantum computers [102, 103].

Machine learning's recent successes revolved around supervised classification such as the ImageNet challenge [3, 104] and, more recently, generative modeling with transformers [10] and diffusion models that are inspired by ideas from non-equilibrium thermodynamics [105]. A straightforward application of supervised classification to quantum physics was explored by Juan Carrasquilla and Roger Melko when they trained a neural network to recognize phases of matter from data [90]. Identifying phase transitions is a fundamentally difficult task, often tackled by tracking an order parameter such as specific heat. In situations where such an order parameter is not apparent, it is not easy to identify and classify phases of matter. Yet using neural networks, it was possible to train an algorithm to classify phases of matter simply from data.

Since quantum states are described by vectors in an exponential Hilbert space, it may appear somewhat surprising that a neural network could easily classify such states without requiring a large number of parameters. However, quantum states are already a subset of the total possible states in the Hilbert space, e.g., density matrices form a convex set [106]. Under more physical constraints, a quantum system may only occupy a small fraction of the total Hilbert space, therefore simplifying their classification or approximation, see Fig. 1.2.

Neural-network quantum states proposed by Carleo and Troyer achieve this feat [41]. Quantum states approximated with different types of neural networks show great promise in tackling the dimensionality problem that a general quantum state description faces [107–110]. The intuition and hope is that automatically learning an efficient representation that captures the essential features of a quantum state can simplify its estimation from data with fewer measurements. Further, it was possible to study the dynamics of a quantum system more efficiently with the approximation of a neural-network ansatz [111].

In the appended Papers I and II in this thesis, we show how the most straightforward approach of representing a quantum state as the output of a neural network can be used for tomography. The goal is to learn the underlying distribution of measurement outcomes on the state from observed data. This problem relates closely to generative modeling in machine learning, e.g., a neural-network-based approach can learn the underlying distribution of pixels that create a human face or generate a scenery from text input [112–114]. In a broad sense, this is also an inverse problem, where we want to estimate the parameters of a model that

uses a density matrix from observed measurement data. In the quantum tomography task, additional constraints coming from physics complicate the problem and have to be enforced. Beyond characterizing quantum states in a quantum computer, we want to solve problems with it.

A quantum computer is used to solve problems by encoding and manipulating information with quantum systems, i.e., running a quantum algorithm. Quantum algorithms are implemented as operations on quantum states. The operations are designed to increase the probability of solution states to a problem. Consider the task of finding the prime factors of a number. It is a computationally difficult problem for classical computers. Shor's quantum algorithm showed one can find prime factors by encoding the problem in a quantum computer using a polynomial number of quantum operations [115]. The quantum algorithm is exponentially faster than the best-known classical approach. Other algorithms implemented in a similar way by using a tunable physical system such as a quantum computer could solve many difficult problems [116].

In order to implement such operations, we need to optimize the control parameters of a quantum device. Such optimization is often posed as an optimal control problem [117]. This is another avenue where machine learning can help [118, 119]. We discuss, in Papers III and VIII, how simple gradient-based optimization with constraints allows us to find parameters that implement a target quantum operation, e.g., unitary operations that generate non-classical states of light or implement logical operations on encoded quantum states. In a quantum device, such an operation is implemented by modulating pulses that affect the controllable part of the Hamiltonian. Pulse-level simulations and modeling are discussed in Paper V. Although we do not go into the optimization of pulses in this thesis and use the black-box tool [120] for optimization, the software developed in Paper V could enable future research into optimizing pulses and low-level simulation of quantum algorithms incorporating various type of noise.

1.4 Quantum computers for machine learning

Another aspect of combining quantum physics with machine learning is using a quantum computer itself for machine learning. A quantum computer can be used as a learnable model, or to compute interesting physical quantities such as generalized susceptibility or nuclear forces with methods inspired by machine learning. We present one such idea in Paper IV with multiple

applications.

Recent demonstrations of the capabilities of quantum computers show other promising applications beyond Shor's factoring algorithm [121], with many of them being heuristics [122]. Variational quantum algorithms (VQEs) represent one such approach [39], where a parameterized set of operations $\mathcal{E}(\theta)$ implemented on a quantum computer finds the solution to a problem by optimizing the parameters θ using a classical computer. The quantum computer is used to compute measurement statistics after applying operations and the classical computer uses the results of these measurements for optimization.

A simple example is finding the minimum of a function that gives the ground-state energy of a quantum system defined by some Hamiltonian. A related problem is finding how the ground state changes as we change the Hamiltonian itself, i.e., computing ground-state gradients as a function of the Hamiltonian parameters. Such gradients also connect to many physical quantities of interest, e.g., susceptibilities and nuclear forces in quantum chemistry. By borrowing an idea from machine learning, we show in Paper IV how such gradients can be computed automatically on a quantum computer using implicit differentiation, thus extending the power of variational quantum algorithms.

Therefore, in this thesis, we will discuss the intersection of machine learning (ML) and (Q)uantum physics, noting that combining ML and Q can lead to very different outcomes which do not commute:

$$[\mathrm{ML}, \mathrm{Q}] \neq 0. \tag{1.12}$$

1.5 Outline of the thesis

This thesis is a compilation of the appended papers discussing various aspects of the merger between quantum physics and machine learning. We will present several machine-learning-inspired approaches to tackle tomography and learning in quantum systems. A crucial difficulty in such approaches is enforcing quantum-mechanical constraints, which we tackle in various ways — from implementing differentiable physics-based models to modifying gradient-based optimization using the idea of optimization on a manifold. We also discuss how Bayesian estimation techniques can help estimate quantum states with noisy data incorporating priors. Lastly, we show how quantum computers used as variational models can be enhanced using an idea from machine learning.

The thesis partly builds upon the licentiate thesis of the author on quantum state characterization with deep neural networks. The introduction sets the context for the work by relating some problems in quantum physics that can be formulated as learning problems. The potential benefits of combining machine learning and quantum physics are outlined with examples of recent successes in applying machine learning to quantum physics problems. A brief discussion of the numerical and experimental difficulty of solving learning problems in quantum physics motivates applying machine-learning methods to quantum physics problems. The rest of the chapters introduce the various theoretical tools used in the appended papers.

Chapter 2 provides an overview of machine learning, focussing on modeling and inverse problems. Machine-learning tasks can often be formulated using generative or discriminative models. Neural networks find extensive use in such modeling, and the chapter discusses how generative and discriminative models can be formulated as neural networks. This discussion is relevant for Papers I, II, VI, and IX, where neural networks find use for tomography and the discrimination of quantum states. The connection to Bayesian approaches for modeling is also discussed to highlight some benefits of a Bayesian viewpoint over using neural-network-based models, relevant to the results obtained in Paper X.

The next step after modeling is the estimation of parameters, which can be performed in various ways using optimization algorithms. Chapter 3, therefore, discusses some of the optimization techniques used in the papers with comments on how to include constraints in the optimization. Regularization and constraints play an important role in machine learning and inverse problems and can be implemented in various ways. Regularization also reduces the need for data and proves helpful in handling noise in estimation tasks. The exact implementation of various constraints during optimization is essential to satisfy quantum-mechanical constraints. The results in Papers I, III, VI, VIII, and X require implementing such constraints and regularization. We discuss gradient-based optimization, Monte Carlo methods, and convex optimization in the rest of the chapter as these methods are used in all the Papers from I to X except in Paper V.

Chapter 4 then discusses how the concepts developed so far apply to learning in quantum systems. Quantum state and process tomography are discussed with their challenges and how machine-learning-inspired methods have tackled some of them. Tomography with neural networks is one such approach discussed in this chapter and Papers I, II, VI, and VIII. The use

of non-neural-network approaches that closely follow the recipe used in machine learning, albeit using physics-inspired ansatzes, is also presented in this chapter. Beyond tomography, the optimization of quantum operations within a unitary ansatz is discussed next, which enabled the experimental results in Papers III and VIII. Implementing quantum operations, e.g., a unitary corresponds to physically driving a quantum system with control pulses that act through a controllable Hamiltonian. Pulse-level simulation of quantum systems is implemented in Paper V with a software package. Optimization of pulses and the simulation of quantum dynamics at the pulse level are essential to correctly incorporate realistic sources of noise and find ways to tackle them. Finally, this chapter presents a simple Bayesian approach to model a quantum state used in the experimental Paper X. The Bayesian approach has several benefits over just estimating parameters by simple gradient-descent, which are highlighted.

Chapter 5 delves into the field of quantum machine learning, discussing topics such as variational quantum algorithms and using parametric quantum circuits as machine-learning models. Gradient computation on a quantum device has its complications, and this chapter discusses ideas such as the parameter-shift rule that allow computing gradients on a quantum computer. Then, we present implicit differentiation, the topic of Paper IV that extends the capabilities of variational quantum algorithms and allows the calculation of interesting physical quantities on a quantum computer, hyperparameter optimization of quantum machine learning algorithms, and a new way of creating entangled quantum states.

Chapter 6 summarizes the results from the appended papers, and Chapter 7 concludes the thesis with an overview of the work and possible directions for future research.

Chapter 2

Machine learning

> "truth ... is much too complicated to allow anything but approximations."
>
> John von Neumann [123]

Digital computers, inspired by ideas from Alan Turing [124] and John von Neumann [125], have become indispensable tools today. Computers execute a fixed set of precise instructions, an algorithm, that solves well-defined problems [126]. However, digital computers struggle with loosely-defined, complex real-world problems that involve interacting with the environment, processing noisy unstructured data, or adapting to new situations. A perfect model of the real world is far too complex to handle. Solving many real-world problems necessitates something more than traditional computer programs.

In a task like driving a car, traditional programming is insufficient due to the many possible scenarios and heavy data processing involved. The program must dynamically assess changing road conditions, process noisy sensor data across different modalities, adapt to weather conditions, and make split-second decisions. Furthermore, pre-programming for new scenarios, e.g., driving a new car for which the computer was not programmed, would be challenging. In contrast, humans can learn driving from experience, employing approximations, and adapt to new situations easily.

The idea of making computers "learn" in a way similar to humans

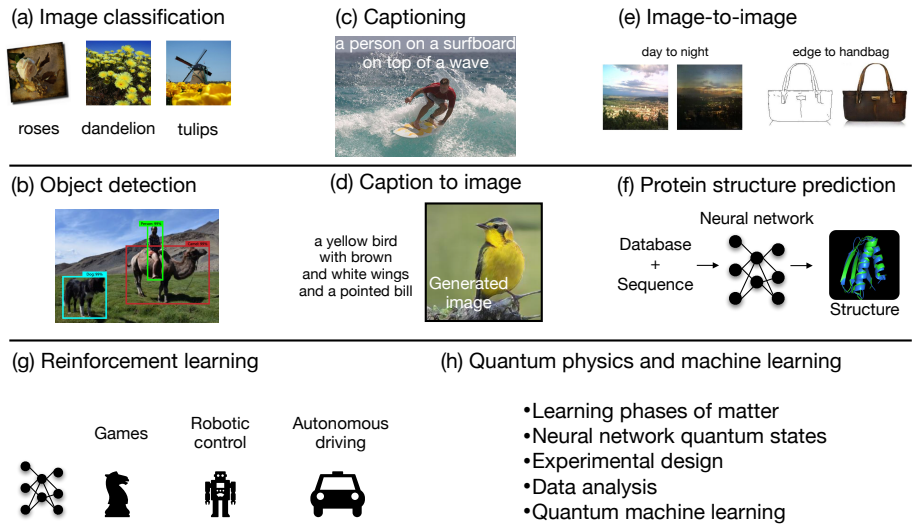


Figure 2.1: Neural-network-based machine-learning applications and their successes. (a) Image classification [127]. (b) Object detection [2, 4]. (c) Generating textual descriptions of images, showcasing a better way than a simple assignment of labels [128]. (d) Generative models can create images from input descriptions [6, 8]. (e) Image-to-image translation highlights how such generative models can create images conditioned on specific inputs [7]. (f) Neural networks also showed superiority in tackling complex scientific problems, e.g., protein-structure prediction [99]. (g) In reinforcement learning, neural networks could lead to computers learning to play games at superhuman levels or solve control problems such as driving [1, 22]. (h) Neural networks have also demonstrated advantages in solving quantum physics tasks, inspiring exploration in this domain [42, 90, 109].

was being explored even before modern computers existed. Alan Turing introduced machine learning as an alternative paradigm for programming computers to emulate human learning, even envisioning the possibility of machines with physical bodies to explore and interact with the world [129]. Despite early skepticism, machine learning and artificial intelligence now excel in problem-solving [130].

Neural networks, specifically deep learning, empowers many modern machine-learning and artificial-intelligence techniques [131]. The success of neural-network-based models over traditional algorithms for pattern recognition and machine learning [132, 133] is speculated to have roots in physical concepts such as symmetry or locality possessed by real-world data [134]. Since 2012, neural networks have revolutionized tasks like text

generation, image processing, conversation, and scientific challenges [131], see Fig. 2.1. These networks model input-output relationships with parameterized functions to solve problems and find extensive applications to model probabilities in machine learning [135] as we discuss in this chapter.

In the 1990s, neural networks were already proposed to solve scientific problems [136]. Initially, they were met with criticism [137] due to the unreliable black-box nature; neural networks seemed excessive against handcrafted methods and models. However, more recently, neural networks have demonstrated success in scientific problems that are difficult to tackle with existing approaches, e.g., fluid-flow predictions using physics-inspired neural networks [138], graph neural networks for material science [139], drug discovery [140], and even tackling grand challenges such as protein folding [13].

In scientific problems, constraints and symmetries play a significant role in simplifying problems, reducing the need for data, tackling noise, and providing interpretability of physically feasible results. We employ several techniques to implement constraints in various applications through Papers I-X, and discuss the main concepts in the next chapter. Ideas around scientific machine learning with neural networks have also lead to emerging new fields such as geometric deep learning [141] and physics-informed neural networks [142, 143]. Neural networks, however, are just parameterized functions that require training resembling solving so-called inverse problems that are often ill-posed [144, 145]. Neural-network training can also be ill-posed [146] and even computationally difficult (NP-hard) [147, 148]. Nevertheless, neural networks have been proposed to solve ill-posed inverse problems by approximating the forward model [149, 150] and work well in practice [134]. Ill-posedness can be tackled through regularization, priors, and constraints on the model or parameters, see Fig. 2.2. Such constraints and regularization are pivotal in the success of neural networks, e.g., convolutions ensure translational invariance in image processing.

In this chapter, we explore the links between modeling, neural networks, and parameter estimation to discuss how quantum physics problems can be tackled with ideas from machine learning.

2.1 Generative and discriminative models

A model simplifies a system, abstracting it to capture core aspects for analysis and predictions. It expresses relationships between variables,

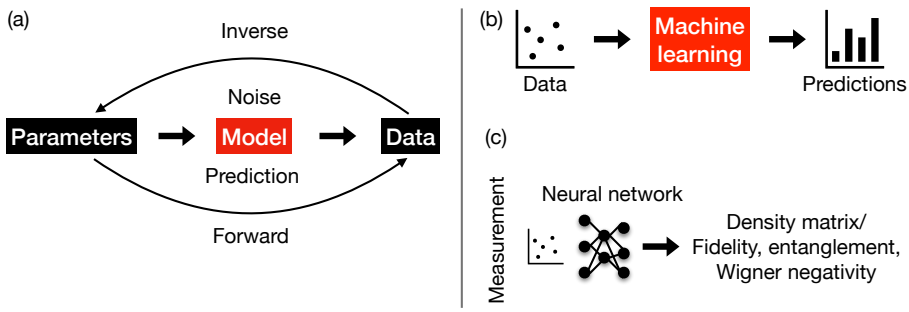


Figure 2.2: Many problems in the scientific domain can be framed as inverse problems. (a) Inverse problems involve parameter estimation from data using a forward model which is often ill-posed due to noisy or insufficient data. Regularization, constraints, and prior knowledge mitigate ill-posedness and the need for large amounts of data. (b) Machine learning offers an alternative to parameter estimation by predicting quantities of interest directly from data with an approximate model. (c) Neural networks can be used for such modeling, e.g., in quantum state tomography, where they can predict or approximate properties of quantum states from measurement data.

learned through functions that allow us to make predictions about physical phenomena. Newton's model of gravitation predicts the speed of falling objects and the orbits of planets. Quantum physics models can predict the behavior of quantum systems, e.g., Bohr's model predicted the emission spectra of atoms.

Models rely on approximations and are often constructed from intuition and observations where a human often proposes the model's functional form, e.g., Kepler's laws resulted from fitting Tycho Brahe's data on planetary motion; Max Planck introduced the idea of the quanta to reconcile theoretical predictions to the data observed in black-body experiments. Machine learning aims to emulate the model-discovery process in a data-driven way. In order to understand data-driven modeling, let us discuss two approaches for modeling in machine learning — generative and discriminative modeling using probability distributions.

Let us consider some input data $\mathbf{x}$, such as the current configuration of a system or an image, and say that we want to predict a property $\mathbf{y}$ for the input. We can consider two key distributions: the conditional probability $p(\mathbf{y}|\mathbf{x})$ and the joint probability $p(\mathbf{x}, \mathbf{y})$. Discriminative models focus on $p(\mathbf{y}|\mathbf{x})$, that label data, e.g., $p(\mathbf{y} = \text{cat}|\mathbf{x})$. On the other hand, generative models learn the joint distribution $p(\mathbf{x}, \mathbf{y})$ that can be used to create new

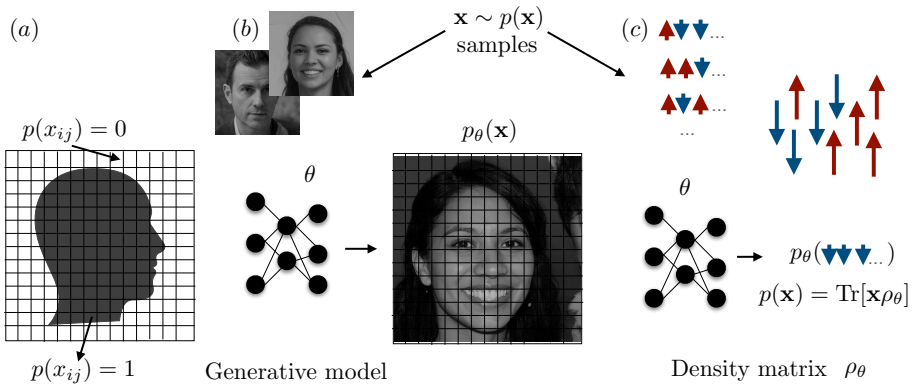


Figure 2.3: (a) Generative modeling of an underlying distribution using samples of data. The probability distribution specifying whether a pixel should be black or white to construct a human face can be complex and intractable. (b) A neural network approximates and learns the underlying probability distribution from samples of data, e.g., human faces. Once the distribution is approximated, new samples can be drawn that resemble human faces. All faces in the above figure are computer-generated; these human faces never existed [151, 152]. (c) In quantum physics, we want to model complex quantum systems to predict probabilities for measurement outcomes or other properties. The density matrix ρ allows us to predict probabilities using the Born rule. Neural-network-based models can be used to learn an efficient approximation for the underlying density matrix (or exactly, as in Papers I and II).

data through sampling, e.g., $\mathbf{x} \sim p(\mathbf{x}, \mathbf{y} = \text{cat})$. Alternatively, we can consider that generative models learn the conditional distribution $p(\mathbf{x}|\mathbf{y})$ while discriminative models approximate $p(\mathbf{y}|\mathbf{x})$.

In most real-world problems, how to model these probability distributions is unclear. Let us consider an image represented as pixel values arranged in a square grid. A combination of values for the pixels could represent an image of a human face. If we consider all possible human faces, there is a particular underlying probability distribution of pixels that can be sampled to generate different human faces, see Fig. 2.3. Writing a mathematical function for this underlying probability density is challenging.

However, machine learning with neural networks can approximate and learn such complex data distributions via training from samples. Human faces have distinct patterns, like an oval shape, a proportional placement of eyes, and other features. Generative models can approximate the underlying distribution, capturing patterns and features to generate new faces that

never existed [151, 152].

In quantum physics problems, the data $\mathbf{x}$ could represent spin configurations or measured parity corresponding to a Wigner function. The target could include classifying the phase or type of quantum state. As we discussed previously, such classification tasks could be challenging due to a lack of strict order parameter for the phase transition or noisy data. However, a neural-network approach could tackle such problems following the discriminative modeling idea, as we show in Papers II and IX.

Similarly, generative models can learn an approximation for the density matrix of a quantum from noisy measurement data, such as Wigner-function values. After training the model using a small set of samples, outcome statistics for any other measurement can be generated through the approximation of the density matrix, e.g., expectation values of the parity function that gives us the full underlying Wigner function, or the full density matrix, as we discuss in Paper X.

At the heart of most such generative and discriminative modeling approaches lie neural networks that are used to model complex relations between data and approximate their distribution.

2.2 Neural networks and universal function approximation

Neural networks consist of interconnected artificial neurons that apply linear and nonlinear transformations to transform inputs $\mathbf{x} \in \mathcal{X}$ to outputs $\mathbf{y} \in \mathcal{Y}$ using a parameterized function $f : \mathcal{X} \to \mathcal{Y}$. The inputs and outputs form the data, $\mathcal{D}$. $\mathcal{X}$ and $\mathcal{Y}$ could be anything from vectors to matrices or arbitrary tensors representing encoding of images, text, video, or measured signals.

If we consider a single neuron with the parameters θ, we can write its output as

$$f(\mathbf{x}; \theta) = \sigma(\theta^T \mathbf{x}), \tag{2.1}$$

where $\mathbf{x} = [x_0, x_1, \ldots, x_N]$ is an input vector and θ is a vector of weights. The term $\sigma(\cdot)$ denotes a nonlinear function called the activation function, which is applied element-wise. The inputs are usually augmented to include a bias term setting $x_0 = 1$. Several such neurons form an output vector $\mathbf{y} = [y_0, y_1, \ldots, y_M]$. A compact way to write this input-output relationship would be to use a matrix of parameters that takes $(N + 1)$-dimensional

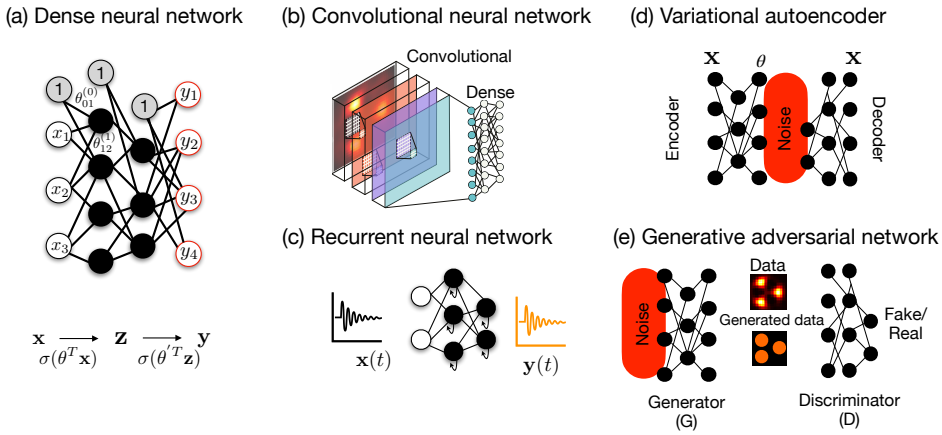


Figure 2.4: (a) A feed-forward, densely connected neural network applies linear and nonlinear transformations to create a learnable function. (b) Convolutional neural networks mimic the human visual cortex by extracting features through spatial filters and use these features, and a feed-forward network in the end, to predict labels. (c) Recurrent neural networks work with sequential data and emulate memory effects through self-connections. (d) Variational autoencoders learn data distributions via an encoder-decoder setup and generate new data resembling the input dataset. An easy-to-sample distribution generates random vectors that are then converted to data by the decoder. (e) Generative adversarial neural networks use competing networks to learn data distributions. The generator produces data from noise vectors and is trained by using a discriminator for assessment.

inputs (including a bias term) and transforms them to M-dimensional outputs as

$$f(\mathbf{x};\theta) = \sigma(\theta \cdot \mathbf{x}), \tag{2.2}$$

where θ is now a matrix.

Neurons can be organized into layers, forming a feed-forward neural network, see Fig. 2.4. A basic neural network with one hidden layer ($l = 1$) connects inputs ($l = 0$) to output $\mathbf{y}^{(l=2)}$ as

$$\mathbf{y}^{(2)} = \sigma\left[\theta^{(1)} \cdot \sigma\left(\theta^{(0)} \cdot \mathbf{x}^{(0)}\right)\right], \tag{2.3}$$

where we assume that the weight matrices are of the right dimensions and account for the bias terms.

Neural networks with a single hidden layer can model any function [153]. This forms the basis for using neural networks to model arbitrary functions and probability distributions. While Ref. [153] used a continuous sigmoid

activation for universal approximation, other activations, like trigonometric ones, can also work via the Stone-Weierstrass theorem [154]. This theorem assures that a polynomial $p(x)$ exists, so that for any x within the interval $[a, b]$, a continuous, bounded, and real function f can be arbitrarily close to $p(x)$, as $|f(x) - p(x)| < \epsilon$ with any error ϵ.

In Ref. [155], the Stone-Weirstrass theorem was used to show that any bounded non-constant activation (even if discontinuous) works for the approximation. In a later work, Ref. [156] showed that standard multi-layer feed-forward neural networks with locally bounded piecewise-continuous activation could approximate any continuous function to any degree of accuracy provided that the activation function is not a polynomial. Extending to multidimensional functions follows naturally.

Universal approximation theorems only guarantee that shallow, single-hidden-layer neural networks can approximate any function, but, in practice, deeper networks with many hidden layers work better [134, 157]. Deeper networks express some functions more efficiently [158, 159], especially in convolutional networks [160]. In some examples, a single-hidden-layer neural network was shown to be unable to realize a three-hidden-layer network [161]. Similarly, a deep convolutional network could not be realized with a shallow network unless the number of nodes in the hidden layer grew exponentially [162]. In Ref. [134], it was shown with a simple demonstration that multiplying n variables required 2^n neurons using a single hidden layer, but $4n$ neurons with a deeper network using $\log_2 n$ layers.

Besides mathematical arguments, physics could also explain why deep learning works well in practice. The data-generation process and the forward model that links parameters to complex data could be seen as a sequence of simple hierarchical steps. The structure of deep neural networks might be more suited to reverse these steps, with each layer distilling information necessary only for the final output [134].

The feed-forward neural network is just one architecture to construct neural-network models. Other formulations of artificial neurons, such as spiking neurons, are also actively researched and pursued [163]. The success of various neural-network architectures can be attributed to specific ways where the model adapts to the nature of data or the problem, see Fig. 2.4. As an example, the transformer architecture, with its attention mechanism that focuses on specific parts of the input, has revolutionized language-related applications [10, 12].

The arguments for function approximation do not discuss the train-

ing and optimization of neural networks. Deep neural networks today have billions of parameters that require optimization [93]. In 1986, the backpropagation algorithm, driven by the chain rule, emerged as a way of computing gradients in a neural network [164] that could then be used for gradient-based optimization. We will cover gradient-based optimization in the next chapter. Before that, let us first explore how neural networks can be used for modeling.

In Papers I and II, we used neural networks to model a quantum system following ideas from generative modeling.

2.3 Modeling with neural networks

In classification tasks, neural networks serve as discriminative models by approximating the conditional probability $p(\mathbf{y}|\mathbf{x})$. We consider samples of labeled data $(\mathbf{x}, \mathbf{y})$, where the target labels are represented as one-hot-encoded vectors. The one-hot-encoding allows a probabilistic interpretation, where elements of $\mathbf{y} = [y_0, \ldots, y_m]$ represent class probabilities, e.g., ($\mathbf{y} =$ [1,0,0,...] says that the probability for the data instance $\mathbf{x}$ to belong to the first class is one, while all other probabilities are zero.

A neural network with parameters θ takes an input $\mathbf{x}$ and predicts a probability vector $\mathbf{v} = [v_0, \ldots, v_m]$ on which a so-called softmax activation is applied as

$$\begin{aligned} \mathbf{v} &= f(\mathbf{x}; \theta), \\ \text{softmax}(\mathbf{v})_m &= \frac{e^{v_m}}{\sum_i e^{v_i}}. \end{aligned} \tag{2.4}$$

The softmax activation ensures that the output $\mathbf{v}$ represents probabilities that sum up to one. The neural network represents a map between data and labels that will output random probabilities at initialization and needs to be trained using data.

An objective function is defined such that the network's output probabilities match the data after training, i.e., we want to minimize a measure of discrepancy between the true (unknown) probability density $p(\mathbf{y}|\mathbf{x})$ and our approximation of it, $f(\mathbf{x}; \theta)$. We can maximize the so-called log-likelihood function by minimizing

$$\mathcal{L}(\theta) = -\sum_{(\mathbf{x},\mathbf{y})} \mathbf{y}^T \cdot \log(\mathbf{v}) = -\sum_{(\mathbf{x},\mathbf{y})} \mathbf{y}^T \cdot \log[\text{softmax}[f(\mathbf{x}; \theta)]], \tag{2.5}$$

where the sum is over all points in our dataset. It is easy to see that if the target and the predicted probabilities are the same, $\mathcal{L}(\theta)$ reaches a minimium. The optimization can be performed using batches of data, usually by gradient descent, as we discuss in Chapter 3. The likelihood function is one possible choice of a loss function that converts neural-network training into a parameter-estimation problem. We discuss more details on this topic in Sec. 2.4.

The use of neural networks for generative modeling is more complicated, and we present some of the approaches below.

2.3.1 Restricted Boltzmann machines

Restricted Boltzmann machines (RBMs) are an early type of generative models, distinct from standard feed-forward neural networks [165]. RBMs have been used for quantum state tomography via the neural-network-quantum-states ansatz [41] and are interesting for modeling quantum systems. An RBM has visible and hidden units ($\mathbf{v} = \{v_1, v_2, ..., v_i\}, \mathbf{h} = \{h_1, h_2, ..., h_j\}$) that give stochastic binary outputs ($v_i, h_j \in \{0, 1\}$). Stochasticity comes from defining the following rule to update the state of a neuron: $h_j = 1$ if the probability

$$p(h_j = 1|\mathbf{v}) = g\left(\theta_{0j} + \sum_i \theta_{ij} v_i\right) \tag{2.6}$$

exceeds a random value sampled from a uniform distribution $U(0, 1)$, where g is the activation function. Visible units update similarly based on hidden units, and the model converges to a Boltzmann distribution at equilibrium:

$$p(\mathbf{v}; \theta) = \frac{1}{Z(\theta)} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h};\theta)}, \tag{2.7}$$

using an energy function

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i \in \text{visible}} \theta_{0i} v_i - \sum_{j \in \text{hidden}} \theta_{0j} h_j - \sum_{i,j} v_i h_j \theta_{ij}, \tag{2.8}$$

where $Z(\theta) = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h};\theta)}$ is the partition function playing a similar role to the softmax activation.

RBMs approximate the underlying probability distribution of data $p(\mathbf{x})$ by modeling it as a parameterized Boltzmann distribution $p(\mathbf{v} = \mathbf{x}; \theta)$. The training proceeds by optimizing a measure of statistical divergence (e.g., KL divergence) between samples generated by the RBM and actual data.

During training, we search for parameters θ that give a high probability for the visible units when set to the data $\mathbf{v} = \mathbf{x}$. However, training RBMs is not easy since, e.g., with gradient-based optimization, the probability's log-likelihood is intractable because of the partition function. Techniques like contrastive divergence [165, 166] enable the training of RBMs but are complicated.

Once trained, new data is generated by some sampling technique, e.g., Gibbs sampling [165]. We can start from a random state v_0 and iteratively update the RBM state until equilibrium, see Fig. 2.5. However, here again, RBMs face difficulty in sampling since the time to reach equilibrium could become large and impractical for larger problems. To handle continuous data with RBMs, we need conversion to a binary encoding or modifications such as Gaussian-Bernoulli RBMs [167]. In contrast to RBMs, approaches like variational autoencoders (VAEs) [168] and generative adversarial networks (GANs) [169] simplify neural-network-based generative modeling by using a latent (noise) variable.

Note that we have not used the label y in the data generation. Therefore, the simple RBM only unconditionally samples $\mathbf{x}$. However, as we see later, it is easy to extend generative models for conditional outputs, e.g., conditional generative adversarial networks (CGANs) [170]. We use CGANs in Papers I and II to generate measurement statistics for quantum measurements in the tomographical reconstruction of quantum states.

We briefly discuss the main ideas here, although only VAEs and GANs are relevant for the results in Papers I and II. Other approaches that use latent variables are normalizing flows [171], score-based [172], and diffusion models [105]. All such models use parameterized neural networks to approximate the underlying data distribution, can be trained with gradient-based optimization, and deal with stochasticity more straightforwardly than RBMs. However, they have significant differences; see Fig. 2.5. Such techniques have not been applied to quantum problems so far, but could have great potential to tackle complex problems in quantum physics, extending the work reported in this thesis.

2.3.2 Variational autoencoders

Variational autoencoders approximate data distributions using latent (noise) vectors $\mathbf{z} \sim p_z(\mathbf{z})$. The latent vectors introduce stochasticity by mapping random elements from an easy-to-sample distribution $p_z(\mathbf{z})$, e.g., a multi-dimensional Gaussian, to the unknown, possibly complicated, target

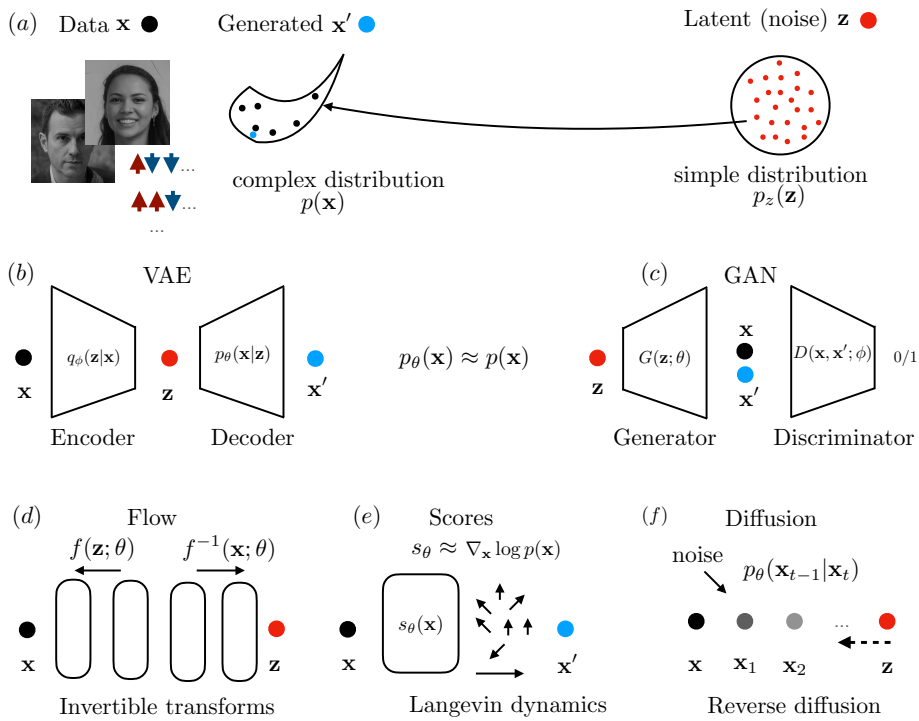


Figure 2.5: (a) Generative modeling with neural networks approximates a data distribution or sample from it using latent (noise) vectors. (b) VAEs maximize the ELBO with an encoder-decoder framework. (c) GANs directly generate samples using a neural network to assess their quality against a dataset. (d) Flow-based models learn invertible transformations of the data to noise that makes the likelihood tractable. (e) Score-based models approximate score function, i.e., the gradient of the log-likelihood that eliminates the intractable partition function. Langevin dynamics generate new data from the approximated score function. (f) Diffusion models reverse a diffusion process that gradually corrupts data by adding noise. The reverse process allows the generation of new data from noise after training.

distribution $p(\mathbf{x})$.

If we consider the joint distribution for data and noise $p(\mathbf{x}, \mathbf{z})$, the probability density for the data distribution can be obtained by marginalization as

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}) d\mathbf{z} = \int p(\mathbf{x}|\mathbf{z}) p(\mathbf{z}) d\mathbf{z}, \tag{2.9}$$

where VAEs assume that the conditional dependence $p(\mathbf{x}|\mathbf{z})$ is given by a so-called decoder neural network [168, 173]

$$p_\theta(\mathbf{x}|\mathbf{z}) = f(\mathbf{z}; \theta). \tag{2.10}$$

Therefore the underlying data distribution is parameterized as

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z}) p(\mathbf{z}) d\mathbf{z}. \tag{2.11}$$

This integral, also called the evidence, is generally intractable due to the geometry of higher dimensions [174, 175]. Intuitively, it would require exploring all possible noise vectors or knowing the high-density regions for $\mathbf{z}$ that contribute to the integral to integrate it efficiently.

We want to use data samples $\mathbf{x}$ to tune θ such that the probability of data in the approximation $p_\theta(\mathbf{x})$ is high. However, since the integral is intractable, we do not have a way to optimize for θ directly. VAEs solve this issue by introducing a so-called encoder function that approximates the posterior

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\mathbf{z}) p(\mathbf{z})}{p(\mathbf{x})}. \tag{2.12}$$

This posterior is also θ-dependent, i.e., $p_\theta(\mathbf{z}|\mathbf{x})$ but is intractable due to the evidence term. In VAEs, the posterior is also modeled by a so-called encoder neural network

$$p_\theta(\mathbf{z}|\mathbf{x}) \approx q_\phi(\mathbf{z}|\mathbf{x}) = g(\mathbf{x}; \phi). \tag{2.13}$$

Now, we have two approximations that take elements from the data space to the latent space and back

$$f : \mathbf{x} \to \mathbf{z}; g : \mathbf{z} \to \mathbf{x}. \tag{2.14}$$

It can be shown that the integral in Eq. (2.9) can be lower bounded using the so-called Jensen's inequality (or, alternatively, using an identity for logarithms $\log x \leq x - 1$). Jensen's inequality says that the expectation

value of a convex function, h, is always lower than or equal to that function evaluated at the expectation value of the argument:

$$\mathbb{E}[h(X)] \leq h(\mathbb{E}[X]). \tag{2.15}$$

We can rewrite the integral Eq. (2.9) as an expectation value as

$$\int p(\mathbf{x}, \mathbf{z}) d\mathbf{z} \quad = \quad \int p(\mathbf{z}) \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{z})} d\mathbf{z} = \mathbb{E}_{p(\mathbf{z})} \left[\frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{z})} \right],$$

and use Jensen's inequality to write the evidence lower bound (ELBO). The ELBO defines a loss function that can be used to train the encoder and decoder [173]

$$\mathcal{L}(\theta, \phi) \quad = \quad p_\theta - D_{\mathrm{KL}}[q_\phi(\mathbf{z}|\mathbf{x}), p_\theta(\mathbf{x}|\mathbf{z})] \leq p_\theta, \tag{2.16}$$

where D_{KL} represents the Kullback–Leibler (KL) divergence. Simply speaking, KL divergence measures how two probability distributions differ from each other. To train the encoder and decoder networks, the KL divergence is minimized using samples of data $\mathbf{x}$, which in turn makes the posterior approximation $q_\phi(\mathbf{z}|\mathbf{x})$ better, as well as maximizes the ELBO. This optimization, therefore, makes the marginal probability $p_\theta(\mathbf{x})$ get closer to the true data distribution.

However, gradient-based optimization for VAEs is not possible directly due to sampling of the latent noise vectors $\mathbf{z}$ that breaks automatic differentiation. This issue was solved by the so-called reparameterization trick, making backpropagation-based training work on VAEs [168]. The reparameterization trick expresses the random variable $\mathbf{z}$ as the output of a parameterized function $\mathbf{z} = h(\mathbf{z}, \epsilon; \phi)$ with an independent random variable ϵ, thereby allowing auto-differentiation to work.

In image-generation tasks, VAEs are susceptible to generating samples that might be blurry and not very sharp, with varying opinions as to why this happens. The use of the KL-divergence loss, simple Gaussian priors, or small latent space dimension are some possible reasons that may lead to VAEs not being able to capture the data distribution well [176]. In this regard, generative adversarial networks perform better and produce sharper outputs in image-generation tasks [177, 178].

2.3.3 Generative adversarial networks

Generative adversarial networks (GANs) [169] take a different approach to learning the map between the latent space and data. Here, a generator

network, $G = f(\mathbf{z}; \theta)$ (similar to the decoder in VAEs), is trained by letting a second neural network, the discriminator $D = g(\mathbf{x}'; \phi)$, evaluate its outputs. The discriminator output is interpreted as the probability for its input $\mathbf{x}'$ to belong to the true data distribution $p(\mathbf{x})$.

The generator and discriminator network can now be trained in an alternating sequence until the generator learns to produce data similar to the actual data distribution such that the discriminator can no longer distinguish generated data from the true data. The parameters θ and ϕ of the two neural networks are optimized alternatingly; first the discriminator parameters ϕ are updated to maximize the expectation

$$\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\Big[\log[D(\mathbf{x};\phi)] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[G(\mathbf{z};\theta)]\Big], \tag{2.17}$$

where $\mathbf{x} \sim p(\mathbf{x})$ are data samples. Then, the parameters of the generator θ are updated to minimize

$$\mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[\log[1 - D(G(\mathbf{z};\theta);\phi]]. \tag{2.18}$$

The conditional generative adversarial network architecture further improves upon the standard idea of adversarial learning by allowing to model the data generation process with the mapping

$$f : \mathbf{z}|\mathbf{y} \to \mathbf{x}, \tag{2.19}$$

where $\mathbf{y}$ is a conditioning variable such as a label. Therefore, conditional generative adversarial networks can model complex conditional maps between different data. The use of the discriminator network to evaluate the generator performance allows possible learning of a non-trivial loss metric. However, since generative adversarial networks do not explicitly model the data distribution and rather learn it implicitly, there are doubts about whether they are truly representative of the data distribution [179].

We have seen that VAEs and GANs do not explicitly learn the probability density for the data distribution $p(\mathbf{x})$ and use a latent noise space only to generate data samples. Variational autoencoders approximate the data distribution, and GANs only generate samples. Several other generative modeling approaches estimate the full probability density function $p(\mathbf{x})$ of data in a creative way to be able to sample from it, which we discuss below.

In Papers I and II, we adapted the CGAN architecture and forced it to estimate the full probability density, i.e., the density matrix of a quantum state. This hinders the scalability of our approach since we demand that the

CGAN learns a density matrix that will have an exponentially increasing dimension as the size of the quantum system grows. However, removing this restriction and directly sample new measurement statistics with such generative models as we saw in the above discussion is possible.

2.3.4 Flow, score-based, and diffusion models

Flows, score-based models, and diffusion models are approaches to generative modeling that aim to approximate a given dataset's underlying probability density $p(\mathbf{x})$ to sample new data. Flows directly approximate $p(\mathbf{x})$ while score-based models learn the gradient $\nabla_x \log(p(\mathbf{x}))$ — the so-called score function. Diffusion models are probabilistic models that directly sample new data points from noise by learning how to denoise data, reversing a diffusion-like process, see Fig. 2.5.

Flows leverage the change-of-variable theorem [171] to transform an easy-to-sample distribution into a more complex data distribution. The main idea is to construct a series of invertible and differentiable transformations that map samples from a simple distribution $\mathbf{z} \sim p(\mathbf{z})$ (e.g., a standard Gaussian) to samples from the target data distribution $\mathbf{x} = f(\mathbf{z}; \theta)$. The generative process involves iteratively applying these transformations to generate data points. The invertible nature of these transformations allows for both data generation and density estimation.

The inverse operation given by $\mathbf{z} = f^{-1}(\mathbf{x}; \theta)$ defines the probability density for the data as

$$p_\theta(\mathbf{x}) = p(\mathbf{z}) \left| \det \frac{d\mathbf{z}}{d\mathbf{x}} \right| = p(f^{-1}(\mathbf{x}; \theta)) \left| \det \frac{df^{-1}}{d\mathbf{x}} \right|, \tag{2.20}$$

where det denotes the determinant. Now, if we consider a series of such invertible functions, we can sequentially convert the simple latent noise sample $\mathbf{z}$ to something more complex by applying the nonlinear bijective functions

$$f_N \circ f_{N-1} \circ ... \circ f_1 : \mathbf{z} \to \mathbf{x}. \tag{2.21}$$

It can be shown that the composition of N such bijective functions is invertible, with the inverse

$$f_1^{-1} \circ f_2^{-1} \circ ... \circ f_N^{-1} : \mathbf{x} \to \mathbf{z}. \tag{2.22}$$

The set of invertible and differentiable functions that map an easy distribution to a complex one makes the likelihood function for the data

tractable. The random variable traverses a path $\mathbf{z}_i = f_i(\mathbf{z}_{i-1}; \theta)$ called the flow. The chain of the intermediate distributions p_i is called a normalizing flow due to movement from a complicated data distribution to a more regular or "normal" form. Different normalizing-flow models can be constructed by choosing the functions f, which need to satisfy two conditions — easy invertibility and computation of the Jacobian determinant [171].

Normalizing flows have the benefit that they provide us with the probability density for the data. Therefore they make it possible to sample new data points, predict the probability for possible data, fill in missing data, and more. They also connect to the narrative of Ref. [134], where deep neural networks provide an excellent tool to model the hierarchical transformation of data using simpler steps to generate complex data.

Score-based models [180] take a different approach when approximating a probability density function $p(\mathbf{x})$ with a parameterized model such as a neural network, $f(\mathbf{x}; \theta)$ [181]. The data distribution modeled as

$$p_\theta(\mathbf{x}) = \frac{e^{f(\mathbf{x};\theta)}}{Z_\theta} \tag{2.23}$$

is learned by maximizing the likelihood $\max_\theta \sum_i^N p_\theta(\mathbf{x}_i)$ where x_i represent individual data points. The normalization (partition function in physics) Z_θ here is intractable, making it difficult to carry out the maximization. In score-based models, instead of modeling p_θ, the score function $s_\theta(\mathbf{x})$ is approximated with a parameterized function $f(\mathbf{x}; \theta)$ and learned. The score function is given by $\nabla_x \log[p(\mathbf{x})]$, which eliminates the normalization constant as

$$s_\theta = \nabla_x \log[p_\theta(\mathbf{x})] = -\nabla_\mathbf{x} f(\mathbf{x}; \theta) - \nabla_\mathbf{x} Z_\theta = -\nabla_\mathbf{x} f(\mathbf{x}; \theta). \tag{2.24}$$

The score function is learned by minimizing the Fisher divergence between the data and model score function:

$$\mathcal{L}(\theta) = \mathbb{E}_{p(\mathbf{x})} = ||\nabla_\mathbf{x} \log[p(\mathbf{x})] - s_\theta(\mathbf{x})||_2^2. \tag{2.25}$$

Since we cannot compute the score for the data $\nabla_\mathbf{x} \log[p(\mathbf{x})]$, a partial-integration trick is used for the minimization of

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \left[\mathrm{tr}[\nabla_\mathbf{x} s_\theta(\mathbf{x}_i)] - \frac{1}{2} ||s_\theta(\mathbf{x}_i)|| \right]. \tag{2.26}$$

Once $\mathcal{L}(\theta)$ is optimized using data samples $\mathbf{x}_i$, we can generate new samples using Langevin dynamics iteratively as

$$\mathbf{x}'_{t+1} = \mathbf{x}'_t + \epsilon \nabla'_x \log(p_\theta(\mathbf{x}')) + \sqrt{2\epsilon} z_t, \tag{2.27}$$

with $t = 0, 1, 2, \ldots, T$ and where $\mathbf{z}_t \sim \mathcal{N}(0, \mathbb{I})$ is a noise vector. In the limit $\epsilon \to 0$ and $T \to \infty$, these iterative steps converge to a sample $\mathbf{x}'$ that follows the data distribution (under some conditions) [182].

Score-based models faced issues in areas of low probability density with few data samples while approximating the score function. These issues were tackled by adding noise perturbations at different scales to the data, allowing learning of the score function in low probability density regions [172].

Diffusion models, specifically denoising diffusion probabilistic models [105], use the idea of constructing a reversible diffusion process that takes a data input $\mathbf{x}$ and slowly corrupts it with noise in discrete steps formulated as a Markov chain $\mathbf{x}, \ldots, \mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_1, \mathbf{x}_0$ [183]. The chain completely transforms the input into a random vector $\mathbf{x}_0 = \mathbf{z}$ and the transition probabilities are given as $p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(x_t; x_{t-1}\sqrt{1-\beta_t}, \beta_t \mathbb{I})$ where we used the Gaussian function $\mathcal{N}(x; \mu, \beta\mathbb{I})$ with mean μ and noise scale defined by β for tractability. A series of positive noise scales define the amount of perturbation for the data at each step: $0 < \beta_1, \ldots, \beta_T$.

The reverse of this noisy process is parameterized with neural networks as $p_\theta(\mathbf{x}_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, \sigma_\theta \mathbb{I}))$ and learned such that it matches the time reversal of the forward Markov chain.

Many other approaches to generative modeling have also been proposed, such as stochastic score-based models that use a stochastic differential equation to approximate the forward and reverse steps [172]. Differential equations have also been used to tackle generative modeling using neural differential equations [184]. The structure of normalizing flows resembles a reversible quantum circuit. Similarly, diffusion models can be imagined as the evolution of a quantum system interacting with its environment that gradually injects noise. Therefore, there are interesting connections to explore between quantum physics and generative models in the future.

In the discussion so far, we have only tackled the modeling aspect of machine learning with some indication of how the parameters of such models can be obtained by minimizing a loss function. In the following sections, we dive deeper into the parameter estimation task, the relation to inverse problems, ill-posedness, and how to tackle it with regularization.

2.4 Parameter estimation and inverse problems

Parameter-estimation problems exist in all fields of physical sciences involving various techniques to obtain parameters of interest defining a model $f(\mathbf{x};\theta)$ over some space with parameters θ using data $(\mathbf{x},\mathbf{y})$. The parameter and data spaces are typically Banach or Hilbert spaces, where a distance measure can be defined using the norm.

The term *inverse problem* encompasses parameter estimation tasks where we have a model for the forward problem relating inputs to outputs. The forward problem is computing the output $\mathbf{y}$ given inputs $\mathbf{x}$ using a model with parameters θ. The inverse problem is finding the solution to estimating model parameters θ such that the model predictions match the expected output $\mathbf{y} \approx f(\mathbf{x};\theta)$. A central question in such problems is the concept of ill-posedness of the problem. Three key elements of this question are (i) the existence of a solution, (ii) the uniqueness of the solution, and (iii) the continuous dependence of the solution on data. These criteria ensure a consistent mathematical model and a reliable description of the physical process connecting the model parameters to data. Problems failing to meet these criteria are known as "ill-posed" problems.

A simple example of the ill-posedness of an inverse problem can be discussed using a linear model given by

$$f(\mathbf{X};\theta) = \mathbf{X}\cdot\theta \tag{2.28}$$

for an input matrix $\mathbf{X}$ of dimension $m\times n$, with a $n-$dimensional parameter vector θ. The inverse problem is computing $\theta = \mathbf{X}^{-1}\mathbf{y}$ given some data vector $\mathbf{y}$ of dimension m. The linear system of equations will always have a unique solution if $\mathbf{X}$ is not rank deficient, i.e., $m = n$ where the rank of $\mathbf{X}$ is $\min(m,n)$. However, if the determinant of $\mathbf{X}$ is zero, we have an ill-posed inverse problem [185].

In addition to ill-posedness, problems can be conditionally ill-posed, where small perturbations in the data lead to large errors in the estimated parameters. In some problems, even if we fulfil the conditions for well-posedness, additive noise given by $\mathbf{y}+\delta\mathbf{y}$ will lead to a significant error in the model parameters since the relative error depends on the condition number of the matrix $\mathbf{X}$ as

$$\frac{||\delta\theta||}{||\theta||} \leq ||X^{-1}|||X|||\frac{||\delta\mathbf{y}||}{||\mathbf{y}||}. \tag{2.29}$$

Therefore even in the simplest parameter estimation or inverse problem formulation involving linear systems, there are several difficulties. Ill-posedness

and ill-conditioned solutions can arise due to noise, imperfect measurements, or insufficient data. It can also occur when the chosen measures generating the data or model are inadequate for accurately capturing the relationship between the data and the model. Various strategies can be employed to tackle such problems, including modifying the model, collecting additional data, or incorporating prior knowledge as parameter constraints.

Machine learning also aims to learn a relationship f that can predict outputs for new inputs after training on a dataset of samples $(\mathbf{x}_i, \mathbf{y}_i)$. However, the model function is often considered a generic class of learnable functions, e.g., a neural network, in contrast to parameter-estimation tasks in the sciences, where the model is known. Recently, using neural networks to map input-output relationships for scientific data has been quite successful even though they represent ill-posed problems [143, 150, 186].

Parameter estimation, inverse problems, or machine learning follow a similar strategy to obtain a model function f using an expected risk. The expected risk measures how well the model predicts the outputs using a positive loss function and minimizing its expectation given by

$$R[f] = \int_{\mathbf{x},\mathbf{y}} \text{loss}(f(\mathbf{x}), \mathbf{y}) dp(\mathbf{x}, \mathbf{y}). \tag{2.30}$$

Minimizing the expected risk becomes challenging since we only have access to a finite set of examples $(\mathbf{x}_i, \mathbf{y}_i)$ and do not know the data distribution. Further, a reasonable choice of the loss function is essential, one that can be tractable and optimized, e.g., a convex function. A least-squares minimization approach [187] is often used where a hypothesis space such as a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ [188] is constructed, and functions that minimize the following expression are desired:

$$\min_{f \in \mathcal{H}} \left[\frac{1}{n} \sum_{i=1}^{n} (f(\mathbf{x}_i) - \mathbf{y}_i)^2 \right]. \tag{2.31}$$

The choice of an RKHS ensures that functions f and g that are close in their norm $\|f - g\|$ are also pointwise close $\|f(\mathbf{x}_i) - g(\mathbf{x}_i)\|$, giving us a sense of distance.

Minimizing the expected risk is one approach to estimating the parameters of a model expressing a relationship between data. Alternatively, consider the task of learning a probability distribution $p(\mathbf{x})$ from samples of $\mathbf{x}$ that give an empirical value for the probability $\mathbf{y} \approx p(\mathbf{x})$. In such

problems, we would like to obtain an approximation for the underlying probability distribution of the data through maximum likelihood estimation.

Thus, learning from examples in machine learning can be seen as solving ill-posed inverse problems, where the model is learned from the data rather than being explicitly designed.

2.4.1 Bayesian estimation

Bayesian estimation is a powerful approach that offers a way to include prior knowledge about the parameters of a statistical model and provides uncertainty estimates for parameters. Bayes' theorem provides the framework for updating beliefs about model parameters θ based on observed data $\mathbf{y}$ as

$$p_{\text{posterior}}(\theta|\mathbf{x}) = \frac{p_{\text{likelihood}}(\mathbf{x}|\theta) \cdot p_{\text{prior}}(\theta)}{p_{\text{evidence}}(x)}, \tag{2.32}$$

with $p_{\text{evidence}}(\mathbf{x})$ the marginal likelihood of the data, given by the integral

$$\int p_{\text{likelihood}}(\mathbf{x}|\theta) p_{\text{prior}}(\theta) d\theta. \tag{2.33}$$

The goal is to infer the posterior probability distribution of the parameters given the observed data representing our knowledge about the parameters and considering the priors. We obtain the full posterior distribution for the parameters instead of a point estimate and therefore can compute uncertainties and make probabilistic predictions. The set of parameters θ^* that has the highest posterior probability [maximum a posteriori estimate (MAP)] can be reported. The MAP matches the maximum likelihood estimate when we assume a uniform prior for the parameters, limiting them to some range, e.g., $\theta \in [a,b]$.

The uniform prior is sometimes referred to as non-informative (although it does contain some information) following a recommendation by Bayes/Laplace. It is not invariant under a reparameterization [189] — if we have no information about a parameter θ, then we should also have no information about any reparameterization such as $\phi = \theta^2$. However, for a one-to-one function $\phi = g(\theta)$, the change of probability theorem gives the probability for ϕ as $p(\phi) = |\frac{d}{d\phi} g^{-1}(\phi)|$, which is not uniform.

A systematic way of constructing non-informative priors that are also invariant to reparameterization was devised by Jeffreys using the Fisher

information defined as the variance of the score function

$$I(\theta) = \mathbb{E}\left[\left(\frac{\partial p_{\text{likelihood}}(\mathbf{x}|\theta)}{\partial\theta}\right)^2\right]. \tag{2.34}$$

The Jeffreys prior is given by $p(\theta) \propto \sqrt{I(\theta)}$ and is invariant under reparameterization. The Fisher information represents the amount of knowledge the data provides about a parameter θ, representing the curvature of the likelihood. If the Fisher information is high, the likelihood function is sharply peaked, and therefore the parameters have a high probability; on the other hand, for a flat curvature, we are more uncertain and hence have a low prior probability for the parameters θ. The influence of the prior is therefore minimized if we have more information about it via the Fisher information.

Bayesian estimation is, therefore, a powerful approach for solving parameter estimation and inverse problems in scientific contexts [190]. Computing the posterior distribution can be challenging due to the intractable evidence term. However, Markov Chain Monte Carlo (MCMC) methods like Metropolis-Hastings or Gibbs sampling can sample from the unnormalized posterior distribution, providing parameter samples directly. We will discuss MCMC methods in the next chapter. Bayesian estimation can also help develop adaptive measurement schemes to guide data collection during estimation.

We use Bayesian estimation in quantum tomography for the results in Paper X to leverage prior knowledge and assumptions. The priors significantly reduced the need for data and could handle noise more effectively. Bayesian estimation was crucial to obtain the results since experimental limitations prohibited collecting a lot of data, and the data had significant noise. In such a scenario, the Bayesian approach was superior to a black-box technique, such as using a neural network or point estimates given by maximum likelihood estimation.

2.4.2 Maximum likelihood estimation

Maximum likelihood estimation (MLE) provides a principled approach to estimating the parameters of a statistical model based on observed data. Many mathematicians, including Lagrange, Bernoulli, Laplace, Gauss, and Fisher, have studied MLE and it has an interesting history marred with attacks, inconsistencies, and triumphs [191]. MLE aims to find the parameter

values that maximize the likelihood function, which measures the probability density of data under an assumed statistical model parameterized by θ as

$$\mathcal{L}(\theta) = \prod_{i=1}^{n} p(\mathbf{x}_i|\theta). \tag{2.35}$$

Instead of directly maximizing the likelihood, it is often more convenient to maximize the log-likelihood function as it simplifies calculations and does not affect the location of the maximum point. The log-likelihood function also converts the problem into a sum over all data points as

$$\ell(\theta) = \sum_{i=1}^{n} \log p(\mathbf{x}_i|\theta). \tag{2.36}$$

To estimate the parameters that maximize the likelihood, we solve the optimization problem

$$\theta^*_{\text{MLE}} = \arg\max_{\theta} \ell(\theta). \tag{2.37}$$

The MLE approach has a connection to the least-squares minimization first discussed by Gauss [192]. It can be shown that maximizing the likelihood of data assuming independent and identically distributed errors is the same as minimizing the least-square error.

MLE, similar to least-square minimization, only provides a point estimate of the parameters that fit the data. However, it is important to note that the MLE approach does not directly provide any measure of uncertainty associated with the estimated parameters. Other limitations such as sensitivity to outliers and biases can become an issue. In cases where prior knowledge or additional information is available, Bayesian parameter estimation methods can provide a more comprehensive framework for parameter estimation incorporating prior knowledge and quantifying uncertainties.

Chapter 3

Optimization with constraints

> "Nature is thrifty in all its actions."
>
> Pierre-Luis Maupertius

The principle of least action is one of the most fundamental laws of nature that is applicable in different scenarios — from deriving Newton's equations to path integrals in quantum physics. It states that the evolution of a system follows paths of least resistance, minimizing the action. Pierre-Luis Maupertius famously stated, "Nature is thrifty", but does nature really work by optimizing over all possibilities before evolving a system? This paradoxical principle also raises questions about predetermination [193].

At the heart of it all is optimization, a crucial technique used across disciplines — from scientific modeling and engineering to machine learning. In many optimization problems, we are only interested in a feasible subset of possible solutions, which leads to constrained optimization. In physical systems, such constraints are enforced naturally, e.g., a spin can only have two possible values. However, while modeling such physical systems and optimizing these models, constraints have to be enforced mathematically.

Parameterizations that naturally enforce constraints are the most straightforward, where techniques such as Bayesian estimation can define a prior parameter distribution that restricts which parameters are considered during optimization. Regularization is another approach that implements soft constraints by penalizing solutions that violate them. Projecting solutions onto a feasible set is yet another approach towards constrained optimization

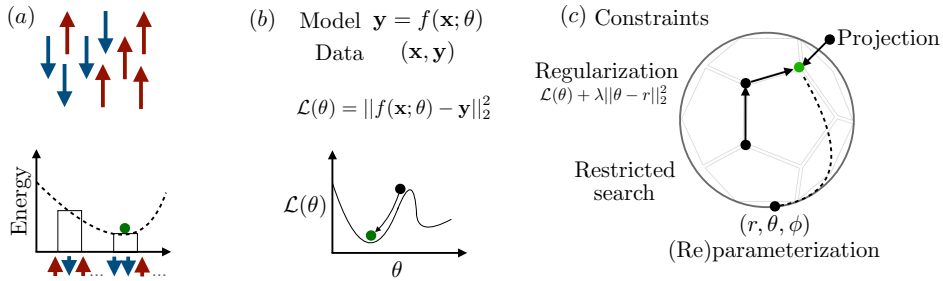


Figure 3.1: (a) Optimization could involve finding the lowest-energy configuration for a spin system. (b) Model fitting by minimizing a loss function using data. (c) Constraints can be handled in various ways, e.g., if the feasible set of parameters lies on a sphere, we could add penalty terms that add a cost to violating the constraint. Projections find the nearest feasible solution after an unconstrained optimization. A reparameterization could transform the constrained optimization into an unconstrained problem. Finally, some techniques only search the space of feasible solutions by cleverly traversing from one feasible point to another.

while some techniques are designed to only explore feasible solutions. All such methods have their advantages and disadvantages, e.g., it might not always be easy to find good parameterizations, or they lead to a more difficult optimization problem. Projections might be costly while only searching the feasible sets could lead to various overheads. In Fig. 3.1, we give a visual explanation of these approaches.

In addition, constraints play a crucial role by restricting the optimization and therefore decreasing the requirement of data in some problems. Regularization and priors can simplify problems to exploit structure, also making the optimization robust. We use different constrained optimization and regularization techniques in Papers I-X (except V) that enforce quantum-mechanical constraints while making optimization robust and data-efficient.

This chapter discusses several such constraints and optimization algorithms. We heavily rely on gradient-based optimization and its modifications. It was humorously quoted that gradient descent can write better code than humans [194], and it remains almost the only available method to optimize models with millions to billions of parameters [195, 196]. However, as ubiquitous as it is, recent research has also explored its limitations [197].

Differentiable programming broadens the scope of gradient descent and neural-network-based problem-solving. We show that not all problems

require neural networks, but still can benefit from the ideas of differentiable programming, regularization, and gradient-based optimization in Papers VI and VIII. In Paper X, we utilize Bayesian parameter estimation with a Monte Carlo method to navigate complex search spaces and obtain samples of parameters that allow uncertainty quantification. The Hamiltonian Monte Carlo method used in Paper X is discussed along with convex optimization since we benchmark our proposed method against compressed-sensing techniques that use convex optimization in Paper VI.

Therefore, this chapter continues the discussion from modeling to finding the parameters of models via optimization under constraints.

3.1 Regularization, constraints, and priors

Parameter estimation optimizes some loss function, $\mathcal{L}(\theta)$, to fit a model to data, see Fig. 3.1. In many problems, the data could be noisy or insufficient, leading to an ill-posed problem. However, if we assume that the model should have sparse parameters, i.e., we want simpler models, regularization can be applied by using the least absolute shrinkage and selection operator (LASSO) [198]:

$$\theta^* = \arg\min_{\theta} \|f(\mathbf{x};\theta) - \mathbf{y}\|_2^2 + \lambda \|\theta\|_{L_1}. \tag{3.1}$$

Here, $\lambda > 0$ is a so-called hyperparameter that defines the regularization strength and $||*||_{L_1}$ is the L_1 norm. LASSO effectively selects sparse models. It is especially useful for ill-posed optimization problems, mitigating noise effects. The right choice of λ is crucial to balance the so-called bias-variance tradeoff, as we see in Paper VIII. High bias with strong regularization overlooks features in the data, while low regularization leads to overfitting noise. However, LASSO has limitations that require consideration [199], e.g., a strong regularization might not recover the true model.

Alternative norms like the k-norm (L_k) are possible penalties:

$$\|\theta\|_{L_k} = \left(\sum_i |\theta_i|^k \right)^{\frac{1}{k}}. \tag{3.2}$$

The L_2 norm defines the so-called ridge regression (Tikhonov regularization [200]) and promotes small but possibly non-zero parameters. The loss gradient is proportional to the current value of the parameter and

therefore with smaller parameters, the regularization effects are weaker. In contrast, LASSO quickly pushes parameters towards zero as the gradient of the regularization term is independent of the current parameter value.

In the Bayesian view, LASSO uses a Laplace prior on the parameters that is sharply peaked at 0:

$$p(\theta) = \frac{1}{2\gamma} e^{\frac{|x|}{\gamma}}, \tag{3.3}$$

where $\lambda = \frac{1}{\gamma}$ matches LASSO's regularization strength. Similarly, the L_2 penalty can be shown to be equivalent to setting a Gaussian prior on the parameters with the regularization strength determining the width of the Gaussian.

In parameter estimation, L_1 regularization (LASSO) is particularly useful when dealing with high-dimensional data, where the number of parameters is large compared to the number of available samples. L1 regularization encourages sparse solutions, effectively performing feature selection and identifying the most relevant variables. This sparsity-inducing property can help identify key predictors or reduce the problem's dimensionality, leading to more interpretable and efficient models.

On the other hand, L_2 regularization (ridge regression) is beneficial when dealing with correlated or collinear predictors. By penalizing large parameter values, L_2 regularization encourages parameter shrinkage and reduces the impact of individual predictors. This can improve the model's stability and generalization performance by reducing sensitivity to noise or small perturbations in the data [201].

We already saw two ways to incorporate prior knowledge and constraints: regularization and setting Bayesian priors. While regularization only represents a soft way to bias solutions, Bayesian priors strongly enforce constraints by defining the parameter distribution. In Paper X, we use a Bayesian approach to reconstruct a density matrix. The priors in the estimation are guided by experimental knowledge and physics-based constraints. Priors represent our beliefs or knowledge about the parameters before observing any data. By choosing appropriate prior distributions, we can encode our understanding of the parameter values and their relationships based on physical laws or empirical evidence.

Another way to strongly enforce constraints is by constructing a parameterized model to enforce physical constraints directly. We will discuss some of the methods to strongly enforce physics-based constraints in the next section.

3.2 Physics-based constraints

In many real-world problems, physical constraints can help guide the parameter estimation process. These constraints are often derived from fundamental principles, empirical observations, or expert knowledge in the field and incorporated into the modeling. We discuss some ideas to implement physics-based constraints below and how they can be combined with data-driven techniques.

3.2.1 Hermiticity

Hermiticity or self-adjointness is a fundamental constraint in quantum physics. One of the basic postulates of quantum mechanics is that a quantum state is described by a complex vector in a Hilbert space acted on by Hermitian operators H. The observable quantities for the state, e.g., energy, are given by the (real) eigenvalues of the operator and any quantum states can be written using the eigenbasis of such an operator.

We can parameterize H as the output of a function, $H(\mathbf{x};\theta) = f(\mathbf{x};\theta)$, such as a neural network. The input $\mathbf{x}$ can be set to a constant value. Then the Hermitian matrix only depends on the parameters θ. While it is not clear how to restrict θ such that the neural network outputs Hermitian matrices, we only need to ensure $H^\dagger = H$. We can do that by transforming the output as

$$H' = \frac{H + H^\dagger}{2}. \tag{3.4}$$

Another way to enforce the Hermitian constraint is to consider the so-called Cholesky decomposition [202] of a matrix given by

$$H' = T^\dagger T, \tag{3.5}$$

where T is a lower-triangular complex-valued matrix with real entries on the diagonal. Instead of parameterizing H, we can now parameterize $T(\mathbf{x};\theta) = f(\mathbf{x};\theta)$ as the output of a neural network. Such a decomposition has been used in several quantum-tomography formulations since Ref. [203]. Since all the steps in these constraint implementations are differentiable, gradient-based optimization works in such problems that have Hermitian constraints. We use this construction in Papers I and II for quantum state tomography with neural networks.

3.2.2 Positive semidefiniteness

The Cholesky decomposition enforces another important physical constraint in quantum mechanics - positive semidefiniteness. Positive semidefinite matrices P obey the constraint $\langle\psi| P |\psi\rangle \geq 0$ for any $|\psi\rangle$. Positive semidefiniteness is an important property of quantum states represented as density matrices. Note that the complex coefficients defining the pure state vectors do not play a role here since taking the conjugate gives $|c_n|^2$, which is always positive.

Therefore, for any state vector ψ_i, the density matrix should obey $\langle\psi_i| \rho |\psi_i\rangle \geq 0$, i.e., it is positive semidefinite.

3.2.3 Unitarity

A complex square matrix is unitary if its inverse is given by the conjugate transpose, i.e., $U^{-1} = U^\dagger$ such that $U^\dagger U = \mathbf{I}$. Unitary matrices are used to represent transformations of quantum state vectors that are linear and preserve the norm. The transformation $|\psi'\rangle = U |\psi\rangle$ makes sure that for a state $|\psi'\rangle$, its coefficients can still be interpreted to give probabilities $\sum_i |c_i'|^2 = 1$. Transformations of quantum states are therefore represented as unitaries, which also can act on mixed quantum states as

$$\rho' = U\rho U^\dagger. \tag{3.6}$$

If we parameterize unitaries with learnable parameters θ (we ignore the data $\mathbf{x}$) that could, for instance, be the output of a neural network, how can we ensure that they remain unitary? In machine learning, such matrices have been used to propose deep neural networks that retain stability against vanishing or exploding gradients since they have an eigenvalue of 1 [204]. We use similar ideas in Paper III for the optimization of quantum gates. Unitary matrices are also the main ingredient for quantum algorithms, variational quantum circuits, and quantum machine learning.

There are several ways to create parameterized unitaries and implement the unitary constraint. In Ref. [204], the authors propose using a set of rotation matrices $\{\mathbf{R}_{ij}\}$ and a diagonal matrix D to write $U = D \prod_{i=2}^{N} \prod_{j=1}^{i-1} \mathbf{R}_{ij}$, where $\mathbf{R}_{ij}$ are identity matrices with the following elements replaced by parameterized functions:

$$\begin{pmatrix} R_{ii} & R_{ij} \\ R_{ji} & R_{jj} \end{pmatrix} = \begin{pmatrix} e^{i\theta'_{ij}} \cos\theta_{ij} & -e^{i\theta'_{ij}} \sin\theta_{ij} \\ \sin\theta_{ij} & \cos\theta_{ij} \end{pmatrix}. \tag{3.7}$$

Other parameterizations are possible, e.g., by using quantum gates. In Paper III, we use the so-called selective number-dependent arbitrary phase (SNAP) gate [205] given by a diagonal matrix and displacement operations to construct arbitrary unitary transformations. Learning unitaries through gradient descent was explored in Ref. [206]. Another approach to implementing the unitary constraint was proposed in Ref. [207], where, starting from a unitary matrix, gradient-based updates project the outcomes to a unitary manifold. The projective approach more generally falls under Riemannian optimization techniques, which were used in Paper VIII and will be discussed in the next chapter.

3.2.4 Complete positivity

Quantum operations are represented by linear transformations of density matrices $\Phi : \mathbb{C}^{n\times n} \to \mathbb{C}^{m\times m}$ that can be generally written as

$$\Phi(\rho)_{ij} = \sum_{kl} \Phi_{ji,lk}\rho_{kl}. \tag{3.8}$$

The tensor $\Phi_{ji,lk}$ is a particular representation for the map; several other such representations exist [208]. For the result of the transformations to be a valid density matrix, Φ must be completely positive. Complete positivity means that the map transforms positive matrices into positive outputs. Choi's theorem on completely positive maps offers one way to ensure this with Kraus operators $\{K_k\}$ [209] that act as

$$\Phi(\rho) = \sum_k K_k \rho K_k^\dagger. \tag{3.9}$$

The Kraus operators can be parameterized in any way, e.g., as the output of a parameterized neural network $f(\mathbf{x};\theta)$, or simply as matrices. However, the Kraus representation only implements complete positivity. The density matrices should also have unit trace. Therefore, in addition to positive semidefiniteness, quantum dynamics should be trace-preserving. The trace-preserving condition is not easy to implement with any parameterization [210] or as a linear constraint. We discuss in the next section how Riemennian optimization can implement the trace-preserving condition.

3.2.5 Differential equations

Differential equations describe the dynamics of variables with complex relationships. They are used for expressing laws of nature and find appli-

cations in various fields, from biology to finance. A classic example is the ordinary differential equation (ODE) describing population dynamics — the Lotka–Volterra model that can be augmented by parameterized functions such as a neural network [211].

Neural ordinary differential equations are written as

$$\frac{d\mathbf{x}(t)}{dt} = f(\mathbf{x}(t), t; \theta), \tag{3.10}$$

where $f(\mathbf{x}(t), t; \theta)$ is a neural network. In such differential equations, choosing the structure of the right side allows us to incorporate physics-based models from theory but also extend them to fill a gap between theory and data, especially noisy real-world data. The training of such models is more involved, requiring advanced techniques that provide gradient information through the adjoint method or implicit differentiation [211].

3.3 Optimization with gradients

Gradient-based optimization is one of the most popular and successful optimization algorithms. It is the workhorse of modern deep learning and various other scientific disciplines. Gradient descent aims to optimize the parameters θ of a function $\mathcal{L}(\theta)$ using gradients iteratively, with $\eta > 0$ being the step size:

$$\theta' = \theta - \eta \frac{\partial \mathcal{L}}{\partial \theta}. \tag{3.11}$$

The vanilla gradient-descent algorithm can be adapted to various situations. If the function $\mathbf{L}(\theta)$ is data-dependent, e.g., the least-squares loss function $\mathcal{L}(\theta) = \sum_i^N ||f(\mathbf{x}_i; \theta) - \mathbf{y}_i||_2^2$, a single gradient update requires calculating gradients for the entire dataset. Stochastic gradient descent (SGD) performs updates for each $(\mathbf{x}_i, \mathbf{y}_i)$, while mini-batch gradient descent makes updates for batches of data with n examples. It has been seen that stochastic gradient descent converges to the minima of convex and non-convex functions with a slowly decreasing step size. However, the simplest approach could get stuck in local minima, have slow convergence, strongly depend on the learning rate, and face a range of other issues, e.g., sparse gradients.

Several improvements to the SGD algorithm have been proposed to solve various issues. Momentum accelerates SGD by dampening oscillations around local minima and pushing it towards updates in previous steps using the vector

$$v_t = \gamma v_{t-1} + \eta \frac{\partial \mathcal{L}}{\partial \theta}, \tag{3.12}$$

with the fraction γ usually set to 0.9. The SGD update rule is modified to

$$\theta' = \theta - v_t. \tag{3.13}$$

In Ref. [212], a number of such modifications to gradient descent are presented. The adaptive gradient method (AdaGrad) considers the structure of the data to adapt the learning rate for different parameters based on the past gradient information. Larger updates are made when gradient information for a variable is sparse and smaller updates otherwise. In AdaGrad, learning rates could diminish rapidly, so the RMSProp method normalizes the learning rate with a term that depends on an exponentially moving average of gradients. We use the Adaptive Moment Estimation (Adam) optimizer [213] for the results presented in several of the appended papers. Therefore we will briefly explain the algorithm below.

The Adam algorithm uses the two first moments of the gradients by maintaining an exponentially moving average of the gradient m_t and the squared gradient v_t given by

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t, \\ v_t &= \beta_2 v_{t-1} + (1-\beta_1) g_t^2, \end{aligned} \tag{3.14}$$

where $g_t = \nabla_{\theta_t} \mathcal{L}(\theta_t)$ represents the gradient at step t and β_1, β_2 are positive coefficients. The exponentially moving average term comes from the idea of exponential smoothing of noisy time-series data x_t, where the next best estimate s_t is considered to be $s_t = \beta x_t + (1-\beta) s_{t-1}$. The initial values are often set to 0 for m_t and v_t which leads to a bias. In Adam, the bias is corrected as

$$m_t' = \frac{m_t}{1-(\beta_1)^t}, \tag{3.15}$$

$$v_t' = \frac{v_t}{1-(\beta_2)^t}, \tag{3.16}$$

such that the update rule is

$$\theta' = \theta - \frac{\eta}{\sqrt{v_t'} + \epsilon} m_t', \tag{3.17}$$

where ϵ is a term for numerical stability usually set to 10^{-8}. The default values for β_1 and β_2 are set to 0.9 and 0.999 but can be optimized with hyperparameter optimization.

In order for gradient-based optimization to work, we require the gradients $\nabla_\theta \mathcal{L}(\theta)$ for very complicated functions and computer programs. The backpropagation algorithm [164] allowed gradient computation for neural networks and recently automatic differentiation tools have made it easier to write computer programs that calculate gradients easily.

3.3.1 Backpropagation

Backpropagation works by propagating the errors from the output layer of a neural network to the input layer in a reverse direction. It can be understood through four fundamental equations that are derived using the chain rule [214].

Let us define the error term $\delta_j^{(l)}$ as the gradient of the loss function with respect to the intermediate output z_j^l of a feed-forward neural network :

$$\delta_j^{(l)} = \frac{\partial \mathcal{L}}{\partial z_j^l}. \tag{3.18}$$

For a mean-squared loss, the error term in the final output layer can be computed as

$$\delta_j^L \propto (\sigma(z_L) - y) \odot \sigma'(z_L), \tag{3.19}$$

where σ represents the activation function, $\odot$ denotes element-wise multiplication, and L denotes the final output layer. To compute the error terms δ_j^l in each layer, we can use the error terms in the next layer δ_k^{l+1} and apply the chain rule:

$$\delta_j^l = \sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l} \delta_k^{l+1}, \tag{3.20}$$

where $\frac{\partial z_k^{l+1}}{\partial z_j^l} = \theta_{kj}^{l+1} \sigma'(z_j^l)$ represents the gradient term and θ_{kj}^{l+1} represents the weights. Once we have the error terms for each layer, we can compute the gradients with respect to the parameters as

$$\frac{\partial \mathcal{L}}{\partial \theta_{jk}^l} = \sigma(z_k^{l-1}) \delta_j^l. \tag{3.21}$$

Backpropagation computes the gradients for a single training instance, but for a loss function that can be averaged over multiple training examples, the gradient computation is straightforward.

3.3.2 Automatic differentiation

The backpropagation algorithm, along with gradient-based optimization, has been a key factor in the success of neural networks in various tasks, including parameter estimation. However, it is important to note that backpropagation is not limited to neural networks and can be applied to any computational graph where the gradients need to be computed efficiently. The concept of automatic differentiation extends beyond backpropagation and plays a crucial role in differential programming. It is a technique that allows the computation of exact gradients in a constant time, unlike symbolic differentiation or numerical computation using finite difference methods [215, 216].

Phil Wolfe made an observation regarding how difficult it is to compute the gradient of a scalar function with respect to n terms and the cost of computing the function itself [215]:

> "If care is taken in handling quantities which are common to the function and derivatives, the ratio [between cost of gradient evaluations and evaluating the function] is usually 1.5, not $n+1$."

Andreas Griewank [215] showed that this observation is a theorem with the upper bound on the ratio as 5. Therefore automatic differentiation makes it possible to compute gradients of a complicated function with the same effort as evaluating the function.

Two modes exist for automatic differentiation — forward and reverse mode (backpropagation). The key is to break down the computation of a complicated function into a sequence of simpler elementary operations such that the chain rule can be exploited for calculating gradients. Forward-mode differentiation works as follows for a series of transformations that take data and parameters to an output for a loss function $(\mathbf{x};\theta) \rightarrow f(\mathbf{x};\theta) \rightarrow g(f) \rightarrow \mathcal{L}(g)$ by evaluating the expressions

$$f(\mathbf{x};\theta), \nabla_\theta f \rightarrow g(f), \nabla_f g \rightarrow \mathcal{L}(g)\nabla_g, \mathcal{L}. \tag{3.22}$$

The total gradient is then computed by taking a product of the intermediate terms. The forward-mode computation, therefore, has to store all the intermediate gradients. If the number of terms in the output is large, forward mode requires large memory. It is, however, fast, as it requires only one forward pass.

Reverse-mode differentiation has an advantage for functions with fewer outputs as it does not store the gradients. It only requires keeping track of

the intermediate outputs and recording how different intermediate terms are connected in a computational graph describing the function and therefore is more efficient [216]. In practice, a combination of forward- and reverse-mode differentiation can be used for arbitrary learning problems, depending on the specific requirements.

3.3.3 Optimization on manifolds

In this section, we present some details about extending gradient descent to manifolds where the parameters have certain restrictions, e.g., they lie on the surface of a sphere. A pedagogical discussion of manifold optimization can be found in Refs. [217, 218]. Here we review the core concepts in a concise way following the discussion in the Supplementary section of Paper VIII. In Fig. 3.2, we present an illustration to explain the core ideas.

The simple gradient-descent algorithm solves the unconstrained minimization

$$\arg\min_{\theta\in\mathbb{R}^n} \mathcal{L}(\theta) \tag{3.23}$$

using a loss function $L : \mathbb{R}^n \to \mathbb{R}$. The search space for the parameters is implicitly assumed to be a Euclidean (flat) space. The optimization is unconstrained because the parameters can take any value in $\mathbb{R}^n$. A Euclidean space is a vector (linear) space that has a norm defined using the inner product. The norm induces a notion of distance between the elements θ and the inner product can be used to generalize the concept of a gradient.

A manifold $\mathcal{M}$ is a set that has a notion of closeness between elements. We can generalize points on Euclidean space to any other set, e.g., the set of orthonormal matrices. A coordinate chart ϕ maps the abstract elements of $\mathcal{M}$ to real numbers $\phi : \mathcal{M} \to \mathbb{R}^n$. Smooth manifolds are those that can be locally mapped to a Euclidean space, i.e., around some point $\theta \in \mathcal{M}$, we can define a composition using a smooth function $g : \mathbb{R}^n \to \mathbb{R}^m$ to map elements of the manifold to real numbers as

$$\phi \circ g : \mathcal{M} \to \mathbb{R}^n \to \mathbb{R}^m. \tag{3.24}$$

We can now define curves on the manifold $c(t)$ that give smooth maps from open subsets of the real line to elements on the manifold $c : \mathbb{R} \to \mathcal{M}$:

$$\phi \circ c : \mathcal{M} \to \mathbb{R} \to \mathbb{R}^n. \tag{3.25}$$

Gradient descent can be regarded as moving along such curves to minimize a cost function. The idea of a directional derivative for arbitrary

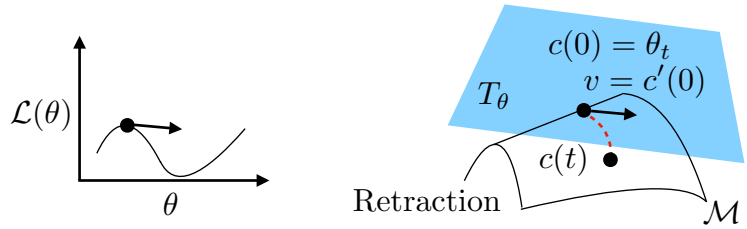


Figure 3.2: Gradient descent on a manifold $\mathcal{M}$ starts by initializing $\theta \in \mathcal{M}$ and calculating the cost gradient $\nabla_\theta \mathcal{L}(\theta)$. Unlike the usual update $\theta' = \theta - \eta \nabla_\theta \mathcal{L}(\theta)$, here, a descent direction $v \in T_\theta$ is determined in the manifold's tangent space. The parameter update follows a curve on the manifold using a technique called retraction. The retraction ensures that the update respects the descent direction while keeping parameters on the manifold.

manifolds can be defined using the differentiability of a curve. The derivative $c'(t) = \frac{dc}{dt}$ can be regarded as a velocity given by

$$c'(t) = \lim_{h \to 0} \frac{c(t+h) - c(t)}{h} \tag{3.26}$$

such that the limit exists. The idea of a curve defines a directional derivative for a function $\mathcal{L} : \mathcal{M} \to \mathbb{R}$ on the manifold as

$$\mathrm{D}\mathcal{L}(\theta)[v] = \frac{d}{dt}[\mathcal{L} \circ c](t)|_{t=0}, \tag{3.27}$$

where $c(0) = \theta$, $c'(0) = v$, and the chain rule applies. Curves and their derivatives allow us to define a linear space — the so-called tangent space given by $\mathcal{T}_\theta \mathcal{M}$ at any point $\theta \in \mathcal{M}$. The tangent vectors in this space can be defined intrinsically without depending on the specific curve c or the choice of the the chart ϕ using the notion of an equivalence set C_θ of smooth curves passing through θ such that for every curve we have $c(0) = \theta$.

In order to formally define gradients that allow us to find directions in the tangent space along which some function increases the most, we have to first consider an inner product on the tangent space. An inner product is a bilinear, symmetric, positive definite function that maps elements from the tangent space to real numbers

$$\langle ., . \rangle_x : \mathcal{T}_\theta \mathcal{M} \times \mathcal{T}_\theta \mathcal{M} \to \mathbb{R}. \tag{3.28}$$

A choice of the inner product that smoothly varies for each point is called a Riemannian metric. By smooth variation we mean that for any $(v, w) \in$

$\mathcal{T}_\theta\mathcal{M} \times \mathcal{T}_\theta\mathcal{M}$, the following map is smooth:

$$\theta \to \langle v, w \rangle_\theta : \mathcal{M} \to \mathbb{R}. \tag{3.29}$$

The metric enables us to define the so-called Riemannian gradient $\nabla_{\theta\in\mathcal{M}}\mathcal{L}(\theta)$ for a function $\mathcal{L}$ on a manifold $\mathcal{M}$ uniquely. To find the Riemannian gradient for a specific manifold and choice of metric, we connect it to the Euclidean gradient of a smooth extension of $\mathcal{L}$ around θ. A smooth extension $\bar{\mathcal{L}}$ on the manifold is given by a map that defines a smooth neighborhood around θ for an embedding of $\mathcal{M}$ as a submanifold of the Euclidean space. The Riemannian gradient can then be evaluated from the definition

$$\mathrm{D}f(x)[v] = \langle v, \nabla_{\theta\in\mathcal{M}}\mathcal{L}(\theta)\rangle_\theta = \langle v, \nabla_\theta\bar{\mathcal{L}}(\theta)\rangle \tag{3.30}$$

where $\nabla_\theta\bar{\mathcal{L}}(\theta)$ is the Euclidean gradient. It is therefore possible to find the Riemannian gradients $\nabla_{\theta\in\mathcal{M}}\mathcal{L}(\theta)$ from Euclidean gradients using the idea of a smooth extension and the inner product.

Once we find the Riemannian gradient, we can iteratively update our parameters along the curve on the manifold that decreases the loss function. Once a descent direction is found, line-search methods find the best step size by satisfying the so-called Armijo-Wolfe conditions for a sufficient decrease of the objective function $\mathcal{L}$ and sufficient increase in the gradient [219].

In Euclidean space, moving along the tangent is straightforward — taking a step along a tangent direction v with stepsize t results in the update $\theta' = \theta + tv$. This notion of taking a step along the tangent is formalized by a retraction on arbitrary manifolds. An exponential map is one example of such a retraction that finds a geodesic $c(t)$ following the tangent vector $v \in T_\theta\mathcal{M}$ such that $c(0) = \theta$ and $c'(0) = v$ where $\theta \in \mathcal{M}$ represents a point on the manifold and $v \in T\mathcal{M}$ represents the tangent.

A retraction can be seen as a continuous wrapping of the tangent space to the manifold. The exponential map in the case of matrix manifolds corresponds to the matrix exponential in some cases. It could become inefficient to compute such exponentials for problems such as those considered in Paper VIII. Therefore we find a retraction using the so-called Cayley curve approximating the exponential map. The manifold that we optimize over is the so-called Stiefel manifold representing a set of orthonormal matrices. The orthonormality ensures that at each step, for the matrix $\mathbb{K}_\theta$ the following condition is satisfied:

$$\mathbb{K}_\theta^\dagger\mathbb{K}_\theta = \mathbb{I}. \tag{3.31}$$

This is precisely the condition on a set of stacked Kraus operators to represent trace-preserving quantum maps. We use this for our quantum-process-tomography technique in Paper VIII.

We have so far focused on using gradient information for optimization which might not be always possible, e.g., for non-smooth functions, when gradient calculation is expensive and noisy, or when the optimization includes probabilities or expectation values that are intractable. In such cases, gradient-free optimization can be useful. One such approach is Markov-Chain Monte Carlo (MCMC) [175], which is used in Paper X to learn the parameters of a Bayesian model for a photoelectron's density matrix. We specifically use Hamiltonian Monte Carlo, which, however, uses gradients again in a clever way.

3.4 Monte Carlo methods

Random search is a simple technique that explores the parameter space and finds a minimum for the cost function simply through sampling. Simulated annealing is another such approach that probabilistically explores the parameter space making jumps depending on some criteria to solve global optimization problems. Monte Carlo methods [220] are often used for such optimization involving sampling. These methods generate random samples of the parameter space, evaluate the objective function, and then use the information to improve the result or for better sampling.

Recall that Bayesian parameter estimation aims to find the posterior probability density of parameters $p(\theta|\mathbf{x})$ modeling a data-generating process given by Eq. (2.32). Analytically evaluating the posterior is often impossible since we cannot evaluate the evidence term in the denominator. However, Monte Carlo methods such as MCMC [175] allow an approximate sampling of the posterior. If we can obtain samples around the mode of the posterior, or the expected value, we find parameters that lead to a high posterior probability fitting the data.

In higher dimensions, computing an expectation value by sampling could become costly [175]. If we were to be efficient, we should only explore relevant parts of the sample space that contributes significantly to the probability density. The high-density regions define parameters that are most likely. In MCMC methods, sampling in high-dimensional spaces happens by constructing a Markov chain of parameters $\theta_0, \theta_1, \ldots$ such that the stationary or equilibrium distribution of the chain corresponds to the

target posterior $p(\theta|\mathbf{x})$. The idea is to construct a chain that gives a high probability to states proportional to the posterior such that sampling from the Markov chain will give us samples from the posterior.

The transition probability $T(\theta_t|\theta_{t-1})$ can be made proportional to the ratio $\frac{p(\theta_t)}{p(\theta_{t-1})}$ (where the conditional dependence on data is implied) such that the intractable evidence term in the denominator cancels out. The random-walk Metropolis algorithm considers such a transition probability where new states are proposed as a random perturbation of the current parameters $\theta_t = \theta_{t-1} + \epsilon$, with a Gaussian noise term. A transition is accepted with the probability

$$\min\left[1, \frac{p(\theta_t)}{p(\theta_{t-1})}\right]. \tag{3.32}$$

In this way, we accept new samples that increase the probability or stay at the same parameter setting. The randomness associated with sometimes not making a transition gives the search a stochastic nature. There is a warm-up phase where the initial samples are discarded. As we run more steps in the chain, we enter a region of high density of the posterior.

The simple random walk faces challenges in high-dimensional spaces, where we could end up in regions of low density. There is an exponentially large number of directions to propose for new parameters in an n-dimensional parameter space. Consider making unit steps for each dimension in two directions $\{+1, -1\}$. The number of ways to propose a new vector is 2^n while the density in the parameter space might be concentrated only in a narrow volume [175]. We, therefore, need an effective strategy to make large jumps in the parameter space and explore it. Hamiltonian Monte Carlo is one such strategy that uses informed jumps for MCMC exploiting the structure of the underlying distribution.

3.4.1 Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC), also known as Hybrid Monte Carlo [175, 221] is a method that uses Hamiltonian dynamics for MCMC. A momentum variable, v, is introduced to define the joint probability density

$$p(\theta, v) = p(\theta)p(v|\theta), \tag{3.33}$$

where the momentum density is generally assumed to be independent of θ, i.e., $p(v|\theta) = p(v)$. The momentum term allows us to move in the parameter

space in a more effective way. If we associate the variable θ with position and define a Hamiltonian $H(\theta, v)$, the canonical distribution over position and momentum is given by

$$p(\theta, v) = \frac{e^{-\beta H(\theta,v)}}{Z}, \tag{3.34}$$

where β is an inverse-temperature term and Z is a normalizing constant. The joint distribution $p(\theta, v)$ therefore defines the probability to find some system with energy $H(\theta, v)$ in state (θ, v).

We are interested in sampling from the parameter distribution and therefore set the Hamiltonian as $H(\theta, v) = U(\theta) + K(v)$, where $K(v)$ is an analogue to kinetic energy and the potential energy $U(\theta)$ is set to

$$U(\theta) = -\log p(\theta|\mathbf{x}) = -\log[p(x|\theta)p(\theta)], \tag{3.35}$$

where we have now ignored the constant term given by the log of the evidence.

The HMC algorithm works by running Hamiltonian dynamics using some method such as Euler or leap-frog [175] and accepting or rejecting a new sample based on the energy term. The equations give the Hamiltonian dynamics

$$\begin{aligned} \frac{d\theta}{dt} &= \frac{\partial H}{\partial v}, \\ \frac{dv}{dt} &= -\frac{\partial H}{\partial \theta}. \end{aligned} \tag{3.36}$$

In contrast to the random Metropolis algorithm, HMC can explore the parameter space more efficiently by using gradient information of the log probability, similar to score-based generative models. These gradients can be obtained by automatic differentiation since they only depend on the likelihood and not the evidence term in Eq. (2.32).

If the kinetic energy term is set to $K(v) = \frac{1}{2}\sum_i v_i^2/m_i$, where m_i is a mass for the i^{th} parameter, we can see that its contribution to the joint density is given by

$$p(v) \propto e^{-\frac{1}{2}\sum_i v_i^2/m_i} \propto \mathcal{N}(0, \sigma_m), \tag{3.37}$$

where σ_m is a diagonal matrix of masses that could be set to identity and $\mathcal{N}(\mu, \sigma)$ is the multivariate Gaussian distribution. Now, we can write the steps for HMC to generate MCMC samples as

- sample a momentum vector $v \sim \mathcal{N}(0, \sigma_m)$
- run Hamiltonian dynamics with $H(\theta_t, v)$ for L steps with some step size η to generate new (θ', v') that is accepted or rejected with the probability

$$\min\left[1, e^{H(\theta,v)-H(\theta',v')}\right]. \tag{3.38}$$

Independently sampling the momentum allows to escape bad directions in the parameter space since if at some point the momentum points towards a bad direction, subsequent resampling of the momentum will cancel its effect. Since the equations are deterministic, i.e., starting from the same position and momentum, we will always reach a fixed final state, the random sampling of the momentum term is crucial. The momentum direction is informed by the gradient of the log-likelihood since

$$\frac{dv}{dt} = -\frac{\partial H}{\partial \theta} = -\frac{\partial \log(p(\theta))}{\partial \theta}. \tag{3.39}$$

Therefore we move towards parameter configurations with higher posterior probability using HMC.

It can be shown that the above construction of MCMC leaves the joint distribution invariant so that the equilibrium statistics represent the posterior probability density that we want to sample from. This condition can be shown by ensuring that the so-called detailed balance equation is satisfied for the probability density under HMC [175].

It is important to note that the hyperparameters L and ϵ must be chosen carefully in HMC. A small number of steps L could lead to random-walk behavior while large L becomes computationally expensive. To address this issue, the No-U-Turn Sampler (NUTS) method [222] was proposed as an extension to HMC that we use in Paper X. The complete NUTS algorithm contains several insights and constructions that improve HMC, but we will only describe the key ideas here.

The crucial step is that NUTS sets the condition to terminate the Hamiltonian dynamics when the distance between new parameters θ_t and the current MCMC sample θ_{t-1} no longer increases. The change in the distance vector $\frac{d}{dt}(\theta_t - \theta_{t-1})/2$ can be shown to be the dot product between the distance and momentum:

$$\frac{d}{dt}(\theta_t - \theta_{t-1})/2 = (\theta_t - \theta_{t-1}) \cdot \frac{d}{dt}(\theta_t - \theta_{t-1}) = (\theta_t - \theta_{t-1}) \cdot v. \tag{3.40}$$

Therefore, when the above dot product starts becoming negative, we are moving back toward the initial state. However, we are not exactly retracing the steps in the dynamics, so time reversibility is not guaranteed [175]. Recall that we also need a transition that obeys a detailed-balance condition such that the target distribution remains unchanged and is truly the posterior. The NUTS proposal solves this with a recursive approach that builds a balanced binary tree of position and momentum values, integrating in the forward and backward direction in time. A termination criterion to stop building this tree is when the momentum vector at the end of the forward and backward trajectory points inwards along the line connecting the position vectors. A more extensive set of such criteria for HMC are discussed and motivated in Ref. [175].

3.5 Convex optimization

Convex (concave) functions are those functions that satisfy the following informal property: the curve of the function lies below (above) the chord connecting any two points on the function. We can frame tomography problems in quantum physics, e.g., reconstructing a quantum process matrix, as the minimization of a linear problem over the set of positive-semidefinite matrices, which is a convex set.

In some problems, we might be interested in minimizing a measure of complexity in addition to fitting data; e.g., in compressed sensing, the sparsity or rank of a signal is minimized. In quantum problems, the rank of a matrix such as the density matrix could represent the purity of the state. Similarly for quantum processes, nearly unitary processes are low-rank. The rank of a matrix is however a non-convex function. Nevertheless, a heuristic that works very well to minimize the rank of a symmetric positive-semidefinite matrix is to minimize the trace, which is a convex function [223]. The compressed-sensing formulation of quantum tomography uses this heuristic to minimize the rank of a density or process matrix while respecting inequality and other constraints, as we discuss in the next chapter.

Let us discuss some basic ideas and algorithms for convex optimization [224] to understand why compressed-sensing formulations that use convex optimization might become computationally expensive for quantum problems, as we see in Paper VI.

Mathematically, convexity is defined for a function $f : \mathbb{R}^n \to \mathbb{R}$ as the

property that, for any $(\theta, \lambda) \in \mathbb{R}^n$ and $t \in (0, 1)$, the following equation is always satisfied:

$$tf(\theta) + (1-t)f(\lambda) \geq f(t\theta + (1-t)\lambda). \tag{3.41}$$

Convex functions have several properties that make optimization easier if the parameters also form convex sets. A convex set from the subset of points in Euclidean space $s : s \in \mathbb{R}^n$ is where a line segment joining any two points in the set lies inside the set. The points of a line and interior points of a circle, sphere, square, or cube are examples of convex sets. However, points on the edge of a circle do not form a convex set since a chord connecting two points on a circle is not part of the circle.

Convex optimization solves the following optimization for a convex set S and a convex function $f : S \to \mathbb{R}$:

$$\min_{\theta} f(\theta); \quad \theta \in S. \tag{3.42}$$

We can show that any local minimum in convex optimization is a global minimum, along with rigorous guarantees on convergence [225]. Also, linear and other convex constraints are easy to implement. If the set of feasible parameters satisfies inequality constraints specified by convex functions g_i, the problem remains convex. Such problems can be formulated as

$$\min_{\theta} f(\theta); \quad g_i(\theta) \leq 0, \ i = 1, \ldots m. \tag{3.43}$$

Assuming that both f and g_i are differentiable, a solution θ^* can be shown to be the global optimum if the so-called Karush–Kuhn–Tucker (KKT) conditions are satisfied [225].

Convex optimization finds widespread use in quantum physics problems. However, as we see in our benchmarks in Paper VI and other works [73], it is not very useful for some quantum problems as the time for optimization grows drastically as the system sizes increase. Ref. [73] reports that going from a two-qubit to a three-qubit problem led to an increased memory requirement of two orders of magnitude and several hours of processing time.

First-order gradient-based methods developed in Paper VI use non-convex formulations of the same problem and are able to scale better. Nevertheless, a discussion of the core principles of convex optimization is essential to understand their limitations. Convex optimization comes with theoretical guarantees and therefore there were attempts to formulate

many machine-learning techniques as convex problems, e.g., convex neural networks [226]. In general practice, however, first-order gradient methods work sufficiently well for non-convex optimization, such as training large neural networks leading to their myriad applications.

Let us look at some basic ideas from convex optimization to understand how they work and the possible difficulties faced.

3.5.1 The simplex method

The simplex method [227] finds the optimal solution to a convex optimization problem by iteratively moving along the edges of the feasible region. It represents a constrained optimization technique where we never leave the feasible set. We do not use the simplex method in any of the appended papers but will go through the basic concept to understand how such a method works to solve constrained optimization.

The simplex method solves the following optimization to which any other linear optimization task can be converted easily:

$$\begin{aligned} &\text{maximize} && c^T\theta \\ &\text{such that} && A\theta = b;\ \theta \geq 0, \end{aligned} \tag{3.44}$$

where $\theta \geq 0$ means that for each element we have $\theta_i \geq 0$. We have m constraints and n parameters. Therefore, the matrix A is $m \times n$.

The set of feasible solutions satisfying the equality constraint is convex and defines a so-called convex polytope — an extension of the concept of a polyhedron to higher dimensions. A vertex of this polytope is an extreme point that intuitively represents a corner or boundary, e.g., the endpoints of a line segment. A point θ is an extreme point if we cannot find two points $\theta_1 \neq \theta_2$ such that $\theta = \lambda\theta_1 + (1-\lambda)\theta_2$ for any $0 \leq \lambda \leq 1$, i.e., the extreme point does not lie between two points in the set.

It can be shown that a maximum in the feasible region is given by an extreme point [224, 225]. If the extreme point is not a maximum, moving away from this extreme point along an edge strictly increases the value of the objective function. The simplex algorithm uses this idea to move along the edges of the polytope to find extreme points with higher objective function values.

The algorithm starts from a so-called basic-feasible solution (BFS) that satisfies the constraints and then updates it iteratively. A new solution is found by updating the elements in the BFS. The so-called entering variable

will be a column index with a positive coefficient in the objective function. Increasing the value of the entering variable from a non-zero value will require changing the values of the current solution to maintain feasibility. A corresponding leaving variable can be selected that keeps the solution feasible. In the revised simplex method, care is taken to make the algorithm memory-efficient compared to the simplex method [227, 228]

Although the worst-case time complexity of the simplex method is exponential in the size of the parameter vector θ, many real-world problems can be solved in polynomial time [229]. However, determining the number of iterations to reach the optimal solution is itself an NP-hard problem, and it is unknown if polynomial-time algorithms exist to solve such linear problems.

In the case of quantum problems, an important constraint is positive-semidefiniteness, e.g., while reconstructing density or process matrices. Such problems are solved using semidefinite programming [230] with methods such as interior point [231, 232], conic solvers, and operator splitting methods [233]. However, as we see in the next chapter, for quantum problems, it is very easy to hit the boundaries of convex optimization problems even for modest-sized quantum systems, e.g., a 5-qubit quantum process tomography task. In such problems, exploiting additional constraints and structure, e.g., a low-rank assumption, can lead to improvements as we show in Paper VI. Similar new ideas have emerged for large-scale convex optimization using manifold optimization [15].

3.5.2 Semidefinite programming

A semidefinite program (SDP) is formulated as

$$\begin{aligned} &\text{minimize} && c\theta \\ &\text{such that } A\theta &= b\ X \geq 0. \end{aligned} \tag{3.45}$$

The positive semidefiniteness (PSD) constraint $\theta \geq 0$ can be seen as an infinite set of linear constraints $z^T\theta z \geq 0, \forall z$.

Therefore the boundary of the feasible set is an intersection of the smooth boundary defined by the PSD constraint and the linear constraints of the problem. Simplex methods will not directly work with such problems.

Interior-point methods can solve such SDPs by first performing an unconstrained optimization on an objective function that includes a so-called logarithmic barrier function [234]. The barrier function penalizes

parameters that violate the equality constraints and updates are ensured to respect the PSD constraint. Interior-point methods have a polynomial complexity; therefore, they are useful for solving large-scale SDPs. The memory scaling is however quadratic. [231, 232, 235]

We use a so-called splitting operator method — the splitting conic solver (SCS) algorithm [233] — in Paper VI, for benchmarks against compressed sensing. The SCS method is a first-order method that can tackle larger problems than interior-point methods. The crucial step of ensuring positive semidefinitenes involves an eigendecomposition that is solved efficiently using a Linear Algebra PACKage (LAPACK), specifically the *dsyevr* method [233] via a projection. This projection step becomes expensive for large matrices. Therefore, even if such semidefinite programs straightforwardly encode learning problems in quantum physics, they could become impractical for use, taking hours for small problems [73]. Instead, we develop a more efficient approach that is not a convex formulation but works well in practice, using gradient-based optimization.

Chapter 4

Learning quantum systems

> "The simulacrum is never that which conceals the truth — it is the truth which conceals that there is none. The simulacrum is true."
>
> Jean Baudrillard

The challenge posed by the simulation of quantum systems with classical computers sparked an interest in building quantum computers, as discussed in Chapter 1. An essential step in this endeavor is characterizing a quantum system by constructing a useful model to analyze the level of control we have over the quantum system. Quantum state tomography estimates parameters defining a model for the quantum state, such as the density matrix ρ. In contrast, quantum process tomography learns how a state transforms, $\rho' = \mathcal{E}(\rho)$, where $\mathcal{E}$ is a representation of the quantum process. In quantum control, the objective is to determine operations that drive a quantum system toward a target.

In all such problems, the central aim is constructing a model for a quantum system and fitting it to (noisy) empirical data. The model can then predict outcomes for new measurements or be used to ascertain the quality of targeted operations. The construction and estimation of such models must adhere to constraints, such as positive-semidefiniteness, set by quantum mechanics. Furthermore, it is desirable to use as few measurements as possible to reduce experimental effort.

However, as discussed in Chapter 1, quantum states and corresponding dynamics are generally represented classically by exponentially growing matrices as the system size increases. Learning such classical models becomes challenging even for moderate-size quantum systems. The challenge is both experimental — we require enough measurements to identify the system uniquely — and computational — the data processing must be done efficiently. In some problems, it may take hours [73] to weeks of data processing [71].

A combination of priors, constraints, and clever algorithms allows us to overcome many challenges associated with the large dimensionality, noise in data, and the computational complexity of learning problems in quantum systems. The structure present in some problems, e.g., a low rank of the density matrix due to it representing a nearly-pure quantum state, can greatly reduce both computational and experimental costs during estimation. This chapter will discuss formulating such learning tasks as machine-learning problems solved by various constrained optimization techniques. The chapter puts in the context the results of all the Papers in this thesis (except Paper IV) with the discussion so far.

4.1 Quantum states and their estimation

We start with learning the parameters of a model representing a general quantum system as the mixture of pure quantum states, the density matrix

$$\rho = \sum_k p_k \left|\psi_k\right\rangle \left\langle\psi_k\right| , \tag{4.1}$$

where $p_k \geq 0$ denote probabilities, and $|\psi_k\rangle$ are complex-valued vectors in a Hilbert space representing pure quantum states. Recall that any state vector can be represented generally using the orthonormal basis vectors $\{|e_n\rangle\}$ of a Hermitian operator in the Hilbert space, as discussed in Chapter 1:

$$\left|\psi\right\rangle = \sum_n c_n \left|e_n\right\rangle . \tag{4.2}$$

The real eigenvalues of these operators e_n represent observable quantities like energy or photon number. At the same time, the complex-valued coefficients c_n give probabilities $|c_n|^2$ that add up to unity for a valid probabilistic interpretation. The density matrix satisfies the condition

$$\left\langle\psi\right| \rho \left|\psi\right\rangle \geq 0 \;\; \forall\psi. \tag{4.3}$$

This condition emerges from the orthonormality of the basis vectors that selects the absolute values of complex coefficients from state vectors $|\psi_k\rangle$, combined with the non-negativity of the probabilities p_k. Further, by rewriting the density matrix in the orthonormal operator basis $\{|e_i\rangle \langle e_j|\}$, we see that the diagonal elements of the density matrix represent probabilities and so should sum up to 1. Density matrices are, therefore, unit-trace positive-semidefinite matrices, making them Hermitian such that $\rho^\dagger = \rho$.

A structure emerges for the density-matrix representation that has parameterization to satisfy both unit-trace and positive semidefinite constraints using the Cholesky decomposition [203]

$$\rho = \frac{T^\dagger T}{\mathrm{tr}[T^\dagger T]}, \tag{4.4}$$

where T is a complex-valued lower-triangular matrix with real diagonal elements. Recall from Chapter 3 that this is one of the simplest ways to enforce constraints — by reparameterizing the problem so that we can run unconstrained optimization over T. In Papers I and II, we use this idea for quantum state tomography using neural networks by letting T be a neural network's output. By rearranging real-valued outputs from a neural network into a complex-valued T, we can represent a quantum state using the parameters of a neural network while ensuring its positive semidefiniteness, see Fig. 4.1.

In order to estimate ρ, similar to learning a probability density, we need samples of measurement outcomes, as discussed in Chapter 1. Measurements can give stochastic outcomes, with the probability of any outcome determined by the Born rule. Note that for a general measurement represented by an operator $\mathcal{O}$, the Born rule defines the expectation value as we show in Chapter 1:

$$\langle \mathcal{O} \rangle = \mathrm{tr}[\mathcal{O}\rho]. \tag{4.5}$$

We are interested in learning the underlying distribution for all measurement outcomes by estimating $p_\theta(\mathcal{O}) = \mathrm{tr}[\mathcal{O}\rho_\theta]$, where we have cast the problem as a generative modeling task faced in machine learning, as discussed in Chapter 2. The parameters θ can be the weights of a neural network such as an RBM [90] or a conditional generative adversarial network (CGAN), as we discuss in Papers I and II.

Quantum state tomography addresses this learning task by choosing measurement settings labeled by $\mathbf{x}$, represented by operators $\{\mathcal{O}_\mathbf{x}\}$, running an experiment to find the empirical expectation values or record samples

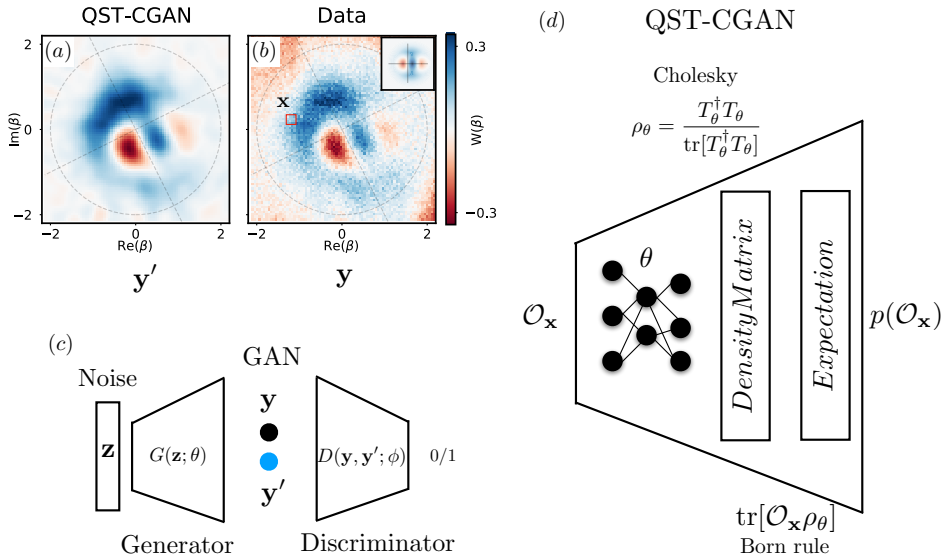


Figure 4.1: Generative models approximate an underlying probability distribution and learn it from data. We develop a method for quantum state tomography with conditional generative adversarial networks (QST-CGAN) in Papers I and II. QST-CGAN learns a representation of an underlying quantum state using experimental data. (a) Reconstruction of expectation values for parity measurements on the phase space by QST-CGAN. The measurement setting is given by a point in phase space specified by a complex number β. (b) The data is noisy and distorted from the ideal target state (inset), even if it contains the main features. Specifying the distortions mathematically in a model is difficult as we might not know the noise sources beforehand. However, a neural-network-based approach could still identify the state to classify it, as discussed in Paper II, or use the raw data's relevant features to estimate the density matrix. While standard generative adversarial networks generate unconditional samples (c), the QST-CGAN in (d) generates samples conditioned on an input measurement setting. This allows us to adapt such a generative model for approximating measurement probabilities $p(\mathcal{O}_\mathbf{x})$. The black-box nature of the generator is simplified by forcing a density-matrix representation internally that obeys quantum-mechanical constraints. The Born rule can then predict expectation values. However, interpretability comes at the cost of scalability as the dimension of the density-matrix representation now restricts us to small system sizes.

of outcomes and then using the data to estimate ρ, which in turn defines $p(\mathcal{O}_{\mathbf{x}})$ through the Born rule.

In most cases, the measurements are given by positive operator-valued measures (POVMs), e.g., measuring the probability of being in one of the orthonormal basis states (ground or excited). The measurements are repeated on copies of the quantum state to obtain a frequency of outcomes due to the probabilistic nature of quantum mechanics. Therefore the data in this learning problem, $(\mathbf{x}, \mathbf{y})$, is given by the measurement operators $\mathcal{O}_{\mathbf{x}}$ and outcomes $\mathbf{y}$ representing the expectation value of the measurement (or samples of outcomes).

We can also view the quantum-tomography task as an inverse problem where we have a forward model given by the Born rule that defines the relationship between model parameters, ρ, and observed data. The forward model is linear as it consists of matrix multiplication, $C_{mn} = \sum_k \mathcal{O}_{mk}\rho_{kn}$, and a trace, $\sum_m C_{mm}$. If we take a single measurement operator with matrix elements $(\mathcal{O}_{\mathbf{x}})_{mk}$ acting on a density matrix with the matrix elements ρ_{kn}, the forward model relating parameters to data is given by the Born rule:

$$\begin{aligned} \mathrm{tr}[\mathcal{O}_{\mathbf{x}}\rho] &= \sum_m \sum_k (\mathcal{O}_{\mathbf{x}})_{mk}\rho_{km} \\ &= (\mathcal{O}_{\mathbf{x}})_{00}\rho_{00} + (\mathcal{O}_{\mathbf{x}})_{01}\rho_{10} + \dots \\ &= \sum_j (\mathcal{O}^f_{\mathbf{x}})_j \rho^f_j . \end{aligned} \tag{4.6}$$

In the last step, we flattened the density matrix to collect its elements into a vector ρ^f and multiplied it with the elements from the appropriately flattened $\mathcal{O}^f_{\mathbf{x}}$.

If we collect the flattened operators for N measurement settings in a matrix by stacking them together as $\mathbf{A} = [\mathcal{O}^f_{\mathbf{x}_1}|\mathcal{O}^f_{\mathbf{x}_2}|\dots|\mathcal{O}^f_{\mathbf{x}_N}]$ and put their corresponding outcome probabilities (or expectation) in a vector $\mathbf{b}$, we can construct a linear system of equations defining the problem:

$$\mathbf{A}\rho_f = \mathbf{b}. \tag{4.7}$$

The term $\mathbf{A}$ is sometimes called the sensing matrix [83], and the solution to the tomography problem is given by the inversion of this linear system of equations. However, linear inversion does not guarantee positive semidefiniteness of the density matrix and might be complicated when there is noise in the data. It is also dependent on the condition number of the sensing matrix $\mathbf{A}$ and the existence of the inverse.

Linear inversion is rarely used for tomography (although we use a hybrid method for QPT in Paper VII, based on linear inversion followed by a projection step). Least-squares estimation is an alternative to linear inversion that works well in practice with noise assumed to be Gaussian [236]. Least-squares estimation solves the minimization

$$\min_{\rho} ||\mathbf{A}\rho_f - \mathbf{b}||_2^2, \tag{4.8}$$

which can be further improved with regularization, leading to sparse solutions and better handling of noise in the data, as discussed in Chapter 3.

However, we still need to project the solution to a positive-semidefinite matrix since this property is not guaranteed by least-squares estimation. There are several proposals to project a matrix to a positive semidefinite one, e.g., by discarding negative eigenvalues or finding the nearest positive-semidefinite density matrix to the least-squares solution. However, such projection steps could become computationally expensive and require solving a sub-problem, as discussed in Chapter 3.

Compressed sensing with convex optimization is another approach to solve the tomography problem defined as the linear inversion task, that greatly reduces the required data [237, 238]. Compressed sensing is a technique to recover large signals from very few measurements assuming low signal complexity, such as low rank or sparsity [237]. We discuss convex optimization techniques in Chapter 3 used for compressed sensing.

In the compressed-sensing picture, we answer the question of estimating a d-dimensional vector ρ_f from N measurements using the sensing matrix $\mathbf{A}$. In general, as soon as the $\text{rank}(\mathbf{A}) = d$, we have a system of linear equations that can give a unique solution to the inversion. It is often the case that ρ_f is an almost pure quantum state and therefore has a low rank. The states may have other structures that reduce their complexity, e.g., low entanglement. Compressed sensing, therefore, recovers a reasonable estimate with far fewer measurement settings than the number of unknown parameters. In Ref. [239], it was shown that for 7-qubit tomography a mere 127 Pauli-basis measurements, each repeated only 100 times, was sufficient for state tomography using compressed sensing.

Instead of optimizing the least-squares objective, compressed sensing solves the convex optimization task

$$\begin{aligned} & & \min & ||\rho||_{\text{tr}} \\ & \text{such that} & \mathbf{A}\rho_f = \mathbf{b}; & \quad \rho \geq 0, \end{aligned} \tag{4.9}$$

where $\rho \geq 0$ is the positive semidefiniteness condition and $||*||_{\text{tr}}$ represents the trace norm.

As we discuss in Chapter 3, it can be shown that minimizing the trace norm minimizes the rank of the matrix. In addition, the formulation of the problem with linear constraints keeps it convex. However, as the system sizes increase, the sensing matrix will become large, increasing the computational cost for convex optimization. As we discuss later for quantum process tomography, compressed sensing becomes a bottleneck even for moderately sized problems, e.g., three-qubit process tomography [73], due to the convex formulation.

An alternative to linear inversion, least-squares estimation, or compressed sensing for quantum tomography is maximum likelihood estimation (MLE) [203]. MLE ensures a positive semidefinite reconstruction by parameterizing the density matrix using the Cholesky decomposition. MLE determines the state by maximizing the likelihood

$$\mathcal{L}(\rho) = \prod_i p(\mathcal{O}_{\mathbf{x}_i}; \rho)^{\tilde{y}_i}, \tag{4.10}$$

where $\mathbf{x}_i$ represent measurement settings and $\tilde{y}_i$ represents the number of times an outcome was recorded. The expectation value is given by $y_i = \frac{\tilde{y}_i}{N}$, using which the log-likelihood can be optimized to find an estimate for ρ:

$$\mathcal{L}(\rho) = \sum_i y_i \log p(\mathcal{O}_{\mathbf{x}_i}). \tag{4.11}$$

In Ref. [240], a simple steepest-ascent method was used to maximize the log-likelihood with an iterative algorithm. In Papers I and II, we compare our neural-network-based estimation methods to this iterative maximum likelihood estimation, as well as the "superfast" MLE method of Ref. [241] that was proposed to escape the issue of slow convergence for MLE using gradient descent on the Cholesky factorization. A gradient-based approach with neural networks, however, learns a quantum state with fewer iterations and data by orders of magnitude compared to the MLE techniques, as the results of Papers I and II demonstrate.

The iterative maximum-likelihood method has a nice guarantee to converge due to the convex nature of the log-likelihood. However, there is no guarantee that every iterative step will increase the likelihood, as shown in Ref. [242] with a counter-example. Therefore it might take a long time to converge. In Ref. [242], a diluted nonlinear iterative algorithm was proposed for faster convergence and guaranteed likelihood increase. Still,

due to the enforcement of quantum-mechanical constraints on the density matrix and implementing a minimization in the space of unconstrained operators, convergence is slow [241].

In Ref. [236], it was possible to develop a fast MLE method for state reconstruction by assuming Gaussian statistics for the measurement noise. The likelihood function is now changed to

$$\mathcal{L}(\rho) \propto \prod_i \exp[-(y_i - \mathrm{tr}(\mathcal{O}_{\mathbf{x}_i}\rho))]^2, \tag{4.12}$$

where we have ignored constant terms. The log-likelihood function now has a form similar to least-squares regression, and maximizing the log-likelihood with the Gaussian noise prior could be related to least-squares minimization, as we also discuss in Chapter 3.

As we discussed before, further improvements to MLE came with applying the "superfast" projected-gradient-descent technique in Ref. [241]. Here, the authors use an accelerated gradient-descent algorithm with a "momentum" that helped to achieve faster convergence. To implement the constraint of positive semidefiniteness and unit trace, the density matrix is projected to the space of valid quantum states at each step of the gradient-based optimization. This is achieved by discarding negative eigenvalues of the estimated density matrix and reconstructing it back from the eigenvectors of the current estimate. With this adaptation, it was possible to perform the state reconstruction of an 8-qubit state within a minute and just hundreds of iterative steps. Although maximum-likelihood-based reconstruction is widely used for state reconstruction, some authors have pointed out its limitations compared to linear regression techniques [243] and even questioned its admissibility [244, 245].

Measurement settings must be chosen carefully to satisfy the so-called informational completeness condition as discussed in Chapter 1 to obtain a unique solution. In simple words, IC entails collecting sufficient information to uniquely identify the quantum state and compute the probability for arbitrary measurements under our assumptions. It is not easy to always determine the optimal measurements, and often over-complete measurements are used, as we discuss in Chapter 1. Prior knowledge about the quantum system, noise in the data, and our formulation of the problem determine the amount of information required for tomography. It is, however, not straightforward to incorporate prior information about complicated noise patterns or the quantum state in the learning algorithm.

In Papers I and II, we proposed a quantum state tomography method that uses conditional generative adversarial neural networks (QST-CGAN). The motivation was that neural networks might be able to learn patterns in the data automatically for a better reconstruction of quantum states, see Fig. 4.1. Patterns that are difficult to write down mathematically and get distorted in noisy data, e.g., Schrödinger cat states consisting of two blobs in the phase space with a pattern between them, could still be recognized by neural networks after training. However as we discuss later in the context of quantum process tomography, neural networks are not always essential, and a more principled approach to include prior information and noise is through a Bayesian estimation.

The QST-CGAN method approximates the probability of an outcome as the output of a generative model. A discriminator network evaluates if such probabilities predicted by the generator match the actual experimental data, therefore, serving as a substitute for the loss function. We evaluated the role of different loss functions in Paper II and studied how to incorporate noise into the model.

The density matrix is the most general parameterization for a mixed quantum state with an exponentially growing number of parameters. Other efficient descriptions could exist for specific classes of states to reduce the number of parameters that require estimation. Consider the example of a Greenberger-Horne-Zeilinger (GHZ) state of three two-level systems, given by

$$|\mathrm{GHZ}\rangle = \frac{|000\rangle + |111\rangle}{\sqrt{2}}. \tag{4.13}$$

If we try to reconstruct the density matrix of an unknown GHZ state, we only need the four density-matrix elements at the corners, given by the coefficients of

$$|000\rangle\langle 000|\,, |111\rangle\langle 111|\,, |000\rangle\langle 111|\,, |111\rangle\langle 000|\,. \tag{4.14}$$

Similarly, consider a class of quantum states such as binomial quantum states written in the Fock basis $\{|n\rangle\}$. These states are a superposition of Fock states with the weights given by the binomial coefficients [88]

$$|\psi_{\texttt{binomial}}\rangle = \frac{1}{\sqrt{2^{N+1}}} \sum_{m=0}^{N+1} \sqrt{\binom{N+1}{m}}\, |(S+1)m\rangle\,. \tag{4.15}$$

The states are parameterized by the integers N and S. If we have such prior information about the state, we can use an efficient representation

of the state and learn a reduced number of parameters that can help us reconstruct that state.

However, forcing the representation biases our reconstruction towards our expectations. This bias could lead to results that are not faithful to the actual experiment and a more flexible ansatz for the state is required. In Papers I and II, we were motivated by the universal approximation capabilities of neural networks to propose them as an ansatz for efficient parameterizations of quantum states.

Other efficient representations such as a matrix product state [246] or a tensor network [109] could significantly reduce the complexity of the problem. Recent work on using a tensor-network method to simulate a 53-qubit experiment using a small cluster of GPUs showed significant success [247]. We can think of these approaches as using priors in inverse problems and regularizations to restrict the search space of parameters. Such parameterizations are interpretable and controlled in contrast to a neural network.

Therefore we attempted to force a density-matrix representation for interoperability in Papers I and II, see Fig. 4.1. Even though this idea performed significantly better than maximum-likelihood methods, the exponentially growing size of the density matrix limits its scaling. It was possible to reconstruct a density matrix with orders of magnitude fewer measurements. Still, we could not ascertain what allowed the neural-network-based approach to perform significantly better than maximum likelihood estimation. It could have been possibly due to the regularization and learning of a non-trivial loss function by a discriminator network.

With the success of Papers I and II, we sought to extend the idea to estimate quantum processes. But we quickly realized that it was a combination of gradient-based optimization, a structured model incorporating quantum-mechanical constraints, and regularization, which were more important than using a complicated neural network.

4.2 Estimating quantum processes

In quantum process tomography (QPT), we want to estimate how quantum states change, i.e., learn the dynamics of a quantum system or characterize a quantum operation, see Fig. 4.2. We can formulate QPT as an estimation task similar to QST by using probes ρ_i and measurements $\mathcal{O}_j$. The outcome of measurements on the transformed quantum state $\rho_i' = \mathcal{E}(\rho_i)$ now depends

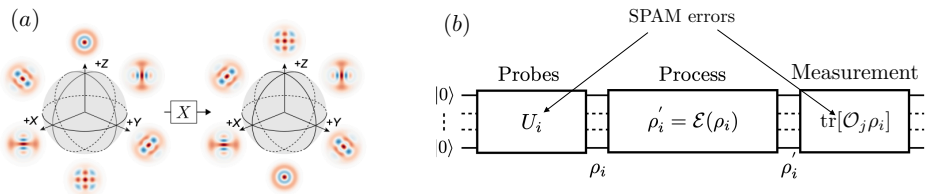


Figure 4.2: Quantum process tomography (QPT) characterizes quantum operations such as (a) transformations of a quantum state that represents some logical information. A bosonic encoding implements computational states by encoding them as orthogonal states of light, e.g., $|2\rangle$, $\frac{|0\rangle+|4\rangle}{\sqrt{2}}$. An operation such as a NOT (X) gate is a physical operation transforming one state to another. (b) QPT proceeds by preparing probe states, applying the target operation, and performing various measurements on the result. State-preparation and measurement (SPAM) errors need to be accounted for to get the true characterization of the target process $\mathcal{E}$. We incorporate estimated SPAM errors in the formulation of the QPT reconstruction in Paper IX.

on both the input probe ρ_i and measurement setting $\mathcal{O}_j$. Let us consider the expectation value (or empirical frequencies for different outcomes) as y_{ij}.

The most straightforward parameterization for $\mathcal{E}$ is a process tensor Φ such that it defines a linear transformation of density matrices:

$$\mathcal{E}(\rho)_{mn} = \sum_{ij} \Phi_{nm,ji}\rho_{ij}. \tag{4.16}$$

The forward model here is $y_{ij} = \mathrm{tr}[\mathcal{O}_j\mathcal{E}(\rho_i)]$. Similar to the linear-inversion formulation of state tomography, if we collect the result of measurements for all probes and operators into a vector $\mathbf{b}$, the QPT task can be formulated as another linear inversion,

$$\mathbf{A}\Phi^f = \mathbf{b}, \tag{4.17}$$

where Φ^f is the flattened representation of the process tensor and $\mathbf{A}$ is constructed using $\rho_i, \mathcal{O}_j$.

Similar to QST, we can use linear inversion, or least-squares optimization, to find the process $\mathcal{E}$ using some representation of it Φ. However, $\mathcal{E}$ has the restriction that it should be

- Completely positive (CP)
- Trace-preserving (TP).

Complete positivity of $\mathcal{E}$ ensures that positive matrices are transformed to positive outputs. The trace-preserving condition means that the trace of the input and output density matrices that the map acts on should be the same. The CPTP conditions ensure the $\mathcal{E}$ represents a valid conversion of one density matrix to another.

One representation of a quantum process is the so-called Choi matrix. The Choi matrix is a linear operator that acts on a tensor product of two Hilbert spaces $\mathcal{H}_{\text{in}} \otimes \mathcal{H}_{\text{out}}$ representing the input and output Hilbert spaces for the process $\rho' = \mathcal{E}(\rho)$. The action of the Choi matrix on a quantum state is given by a partial trace operation:

$$\mathcal{E}(\rho) = \text{tr}_{\mathcal{H}_{\text{in}}}[(\rho^f)^T \otimes \mathbb{I})\Phi]. \tag{4.18}$$

The process $\mathcal{E}$ is CPTP if the Choi matrix is positive semidefinite and the following partial-trace condition is satisfied [209, 248]:

$$\text{tr}_{\mathcal{H}_{\text{out}}}[\Phi] = \mathbb{I}. \tag{4.19}$$

The Choi matrix allows us to use projections [248, 249] to obtain an estimate that might not be physical and then find a CPTP estimate, e.g., for CP, simply discarding negative eigenvalues of the Choi matrix works as

$$\Phi_{\text{CP}} = V \max(0, D)V^\dagger, \tag{4.20}$$

where $\Phi = VDV^\dagger$ represents the singular-value decomposition. In Ref. [249], a new projection method was proposed, called the hyperplane intersection projection, which showed promising advantages over other projection methods. However, most such approaches have to solve linear sub-problems such as eigendecomposition that scale poorly with the size of the problem. Other representations such as using the Pauli Transfer Matrix (PTM) encode the CPTP constraint differently.

The Choi matrix also allows the QPT objective to be formulated as a compressed-sensing problem that can be solved with convex optimization [79]. In many QPT problems, we can assume a low rank for the quantum processes, e.g., a near-unitary operation. Compressed sensing can allow efficient reconstructions exploiting the low-rank structure by solving

$$\begin{aligned} &\min ||\Phi||_{\text{tr}} \\ \text{such that} \quad &\mathbf{A}\Phi^f = \mathbf{b}; \qquad \Phi \geq 0, \;\; \text{tr}_{\mathcal{H}_{\text{out}}}[\Phi] = \mathbb{I}. \end{aligned} \tag{4.21}$$

The minimization of the norm indirectly minimizes the rank of the matrix and the optimization problem can be readily solved by some convex optimization technique. However, the size of the problem quickly becomes very large, e.g., for a 5-qubit tomography, the Hilbert space for the density matrix is 32×32. The Choi matrix is therefore 1024×1024. If we flatten the Choi matrix, the dimension of Φ^f becomes $\approx 10^6$. Such large problems cannot be easily tackled and present computational challenges.

In Ref. [73], it was noted that going from a two-qubit case to three qubits made the computational time for convex optimization go from seconds to several hours. In our benchmarks for Paper VI, we could only tackle two-qubit QPT problems with compressed sensing. In Paper VII, a linear inversion followed by a projection worked better.

The Choi matrix assumes a full-rank process. However, Choi's theorem on complete positivity [209] also shows that a different representation exists. The so-called Kraus operators form an equivalent representation of quantum processes where the rank can be fixed. Kraus operators $\{K_i\}$ act on a density matrix as

$$\mathcal{E}(\rho) = \sum_i K_i \rho K_i^\dagger. \tag{4.22}$$

The Kraus representation guarantees the CP condition by construction. The TP condition is satisfied if

$$\sum_i K_i^\dagger K_i = \mathbb{I}. \tag{4.23}$$

The maximum rank of the corresponding Choi matrix gives the number of Kraus operators. In the worst case, for a full-rank representation, the Kraus representation has as many parameters as the Choi matrix with the rank being the dimension of the Choi matrix. But in many QPT problems, we deal with near-unitary quantum operations with a low Kraus rank. Therefore we can assume only a few Kraus operators in our parameterization for a quantum process and estimate it from data.

The main issue with this approach comes from the TP condition, which no longer can be guaranteed as simply as dividing by the trace in the case of a density matrix's trace-normalization condition. We first used a parameterization of Kraus operators given by a generative neural network. But it was unclear how to restrict the neural-network weights so that the TP condition is satisfied for the Kraus operator. The solution was to modify the optimization from simple gradient-descent to manifold optimization,

recognizing that the set of Kraus operators respecting the TP condition form the so-called Stiefel manifold.

We found that by using manifold optimization, we could use the Kraus parameterization directly without using a neural network. In Paper VI, we show how manifold optimization, discussed in Chapter 3, allows an efficient QPT protocol, where we can control the rank of the process simply by selecting the number of Kraus terms.

Moreover, we can also use all the tricks of stochastic optimization by writing a loss function that only takes in batches of the data and optimizes the loss function iteratively. In this way, we do not have to deal with large matrices as in the straightforward convex formulation used in compressed sensing. Adding an L_2 penalty term in the loss function further allowed us to regularize and deal with noisy data giving sparse solutions.

In Paper VIII, the GD-QPT technique was used to perform tomography of a physical operation on a Hibert space of 32 — one of the largest we found in the literature. It allowed insights into the full space of possible transformations that a quantum system could undergo beyond the reduced encoded subspace.

We can also turn the QPT idea to estimate parameters of gates that lead to quantum states of interest, e.g., the bosonic encoded states in Paper III. In Ref. [206], unitary learning with gradient descent was explored. We take inspiration from the same to optimize the so-called selective number dependent arbitrary phase (SNAP) gates in Paper III. The SNAP operation along with displacement allows universal control of a resonator. If we parameterize them and write a loss function representing the fidelity of a quantum state to some target,

$$\mathcal{L}(\theta) = F(\mathcal{E}(\rho), \rho_{\text{target}}), \tag{4.24}$$

we can perform numerical optimization with gradient descent to learn the gate parameters. We obtained a very efficient series of gates using only three blocks of SNAP and displacement operations, allowing the generation of very interesting non-classical quantum states of light in Paper III.

The learning approaches discussed so far do not provide uncertainty estimates for quantities computed of the state or process. Sometimes even the estimated quantities such as fidelity depend highly on the regularization strength. As we see in Paper VIII and discuss in Chapter 3, the choice of the regularization strength becomes important as a strong regularization would ignore essential features in the data. In contrast, no regularization would overfit to noise.

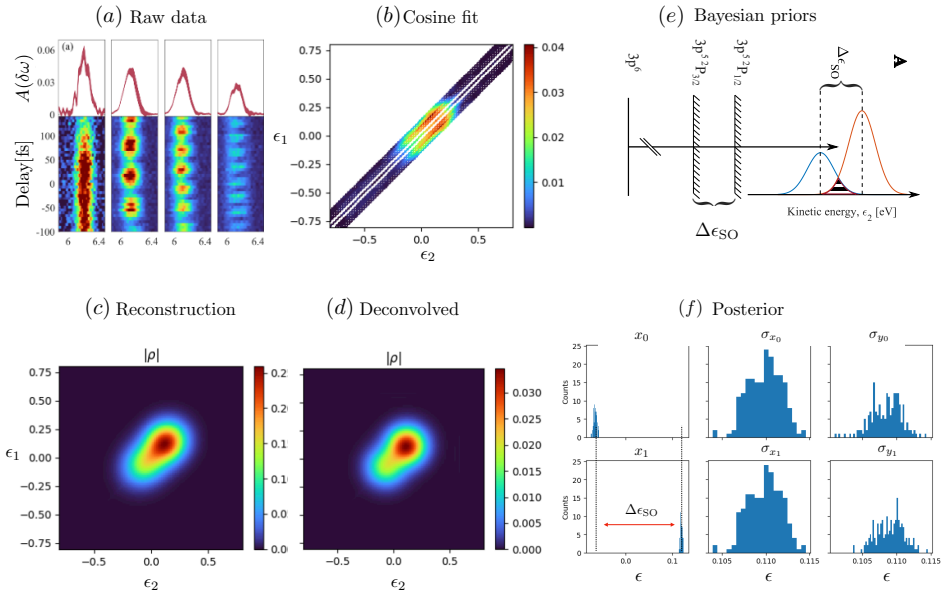


Figure 4.3: Bayesian estimation of the density matrix for a photoelectron. We focus on reconstructing the magnitude of the density matrix $|\rho|$ from a partial estimate using prior information. (a) Raw data is converted to sub-diagonals through a cosine fit. (b) We only have access to a few sub-diagonals and wish to reconstruct the full matrix. (c) Bayesian estimation reconstructs the full matrix under a smooth model using Gaussian functions. (d) The underlying density matrix is estimated using a forward convolution in the model instead of deconvolving the result in (c). Prior knowledge from physics, e.g., the spin-orbit coupling determining the separation between peaks, can be enforced easily within the Bayesian model. (f) We obtain parameter samples that allow the calculation of uncertainties in an interpretable way.

In order to compute uncertainties, in Paper VII, we used bootstrapping [250] that required repeating the reconstruction algorithm many times on re-sampled data. A more principled way to obtain uncertainty estimations and set reasonable priors is using Bayesian estimation. The Bayesian approach allows us to incorporate both noise in the data and set reasonable priors to obtain uncertainty estimates based on the assumptions made, e.g., the regularization strength.

4.3 Bayesian quantum tomography with priors

Bayesian techniques allow the inclusion of prior information and provide not only a parameter estimate but also uncertainty in the parameters. Bayesian quantum tomography can allow an optimal estimation of quantum states that include priors [244, 251].

In Paper X, we applied Bayesian estimation to a tomography task that was too complicated at many levels to be solved easily via linear inversion, or maximum likelihood estimation. The problem was determining the density matrix of a photoelectron [252]. The density matrix is now continuous, $\rho(\epsilon_1, \epsilon_2)$, represented in an energy eigenbasis. In an experiment, we obtain an indirect estimate of parts of the density matrix and wish to find the underlying complete representation of $\rho(\epsilon_1, \epsilon_2)$, see Fig. 4.3.

The measurement data itself is a convolution of the true underlying density matrix elements with noise. None of the traditional tomography techniques suit such a problem where only indirect estimates through a convolution of the data are available. We also operate in a regime where we do not have informationally complete measurements and the solution to the problem is not unique — any arbitrary filling of the missing elements is possible.

However, several assumptions could be made to alleviate the issues in this task. The absolute value of the density matrix for the system is smooth and has Gaussian-like structures for specific situations, e.g., photoelectrons emitted from He. In some cases, theoretical calculations suggest multiple peaks and specific relationships between the peaks, e.g., dictated by some physical consideration such as spin-orbit coupling [252].

In order to incorporate these assumptions into the reconstruction process, address noise effects, and derive uncertainty estimates, we formulate quantum tomography a Bayesian estimation task. We reconstruct the magnitude of our density matrix using a mixture of two-dimensional Gaussians

$$|\rho(\epsilon_1, \epsilon_2)| = \sum_i A_i G(\epsilon_1, \epsilon_2; \vec{\mu}_i; \vec{\sigma}_i), \tag{4.25}$$

where G represents a two-dimensional Gaussian with centers specified by vectors $\vec{\mu}_i$ and widths along two axes given by the vector $\vec{\sigma}_i$. The experimental data gives us the magnitude of the density matrix for different (ϵ_1, ϵ_2). These are therefore our measurement settings, and we have access to an indirect value of the expectation at several (ϵ_1, ϵ_2).

Now we have reduced the problem to parameter estimation of a set of Gaussian functions given samples. In this particular example, experimental considerations make it easier to sample slices of the sub-diagonal elements. We also assume additive Gaussian noise in the measured data with a known noise variance.

Therefore in the Bayesian model, we are able to directly include the various known sources of errors and noise directly in the reconstruction. In addition, the assumption of Gaussians strongly restricts our estimation to smooth bell-shaped curves that we expect from theoretical considerations.

We perform a Markov Chain Monte Carlo (MCMC) exploration of the parameter space to obtain posterior samples of the Gaussian parameters. The Hamiltonian Monte Carlo (HMC) method is used, enhanced by the No-U-Turn Sampler (NUTS) as we discussed in Chapter 3. It might be possible in this particular example to have a tractable computation of the posterior since we use Gaussian functions; however, there is a catch to the simple picture.

The sampled data corresponding to $|\rho(\epsilon_1, \epsilon_2)|$ actually represent the convolution of the true signal with the point-spread function (PSF) of the detector. Therefore it needs to be deconvolved first before assuming that they are the density matrix elements. This is a complicated deconvolution procedure that requires a careful inversion of a convolution operation between a noisy signal where the convoluted PSF itself could have uncertainty. Instead of performing a deconvolution, we can simply include a convolution in the forward model while defining the likelihood function.

As a result of this procedure, we are able to reconstruct the density matrix of a photoelectron for the first time where other tomography techniques would have been difficult to apply. Prior knowledge about the system is incorporated by selecting parameter distributions carefully. We know the range for each parameter, e.g., the centers of the Gaussians, and can set them accordingly. In addition, it is possible to include other priors dictated by the physics of the system. One such prior is that in the case of photoelectrons ejected from argon, the magnitude of the density matrix contains two peaks with one of the amplitudes being twice the other. The centers of the two peaks are separated exactly by the so-called spin-orbit (SO) coupling value [252]. We can include these priors by allowing one of the centers to fall within a certain range $x_0 \in [a, b]$ and the other center to be sampled from a Gaussian with its mean at $x_0 + \Delta\epsilon_{\mathrm{S.O.}}$. Again, the width of this Gaussian will determine our confidence in the physical assumptions

made.

A benefit of the Bayesian procedure is that we can exactly quantify the uncertainty in the estimate that corresponds to our belief in the parameter space (using the priors), and the noise in our measurements (by setting the scale of the noise in the likelihood). It is therefore greatly reducing the need for data in a problem that would otherwise demand many more measurements if we looked at it from the point of view of linear inversion.

Note that it is also not possible to solve this problem with the application of neural networks that can fill in the density matrix in any way possible without respecting the physical constraints and priors. Another possible way of filling was using Gaussian processes; however, it also does not give the fine-grained control that we have from an explicit model.

Even if we implemented the symmetry constraints that make sure the matrix is Hermitian, positive semidefiniteness is not directly guaranteed. A consequence is that off-diagonal elements might be larger than the corresponding diagonal ones, so we can set a prior that ensures that the off-diagonal terms are smaller than the main diagonal. This is achieved by defining a relationship between the widths of the Gaussian along the diagonal and the off-diagonal.

We can also model the complex phase of the density matrix. In this case, theory tells us that the complex part is not Gaussian, but modeled as constant planes. Once we can model the complex part, we can ensure positive semidefiniteness by a final projective step after an estimate is found. However, the best way would be to be able to implement the PSD constraints in the modeling itself. This would mean defining a general distribution of quantum states for such a problem that enforces positive semidefiniteness [251].

We have so far discussed estimation tasks for quantum problems using classical optimization and modeling. In all the problems, we approximate the quantum density matrix (or process matrix) with a classical parameterized function that we can tune, $\rho = f(\mathbf{x}; \theta)$. However, quantum computing can allow us to tune a quantum function $q(\mathbf{x}; \theta)$ and use it as a model to solve problems. This idea leads to the emerging field of quantum machine learning where a quantum computer is a tunable model instead of a classical function, e.g., a quantum neural network [253].

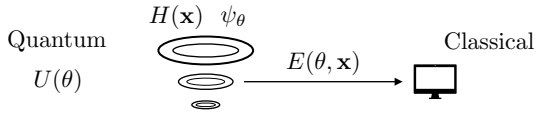
Chapter 5

Quantum machine learning

> "According to strong AI, it is simply the algorithm that counts. It makes no difference whether that algorithm is being effected by a brain, an electronic computer ..."
>
> *The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics*
> Roger Penrose

The algorithms that have been most successful for machine learning in recent times use neural networks and gradient-based optimization to learn. Even if such algorithms are successfully explaining jokes, answering deep philosophical questions, or generating art, at the heart of their successes lies the computational advances brought by specialized hardware. Even more interesting is the fact that most of the computation in modern machine learning with neural networks is matrix multiplication and it is precisely the ability of graphics processing units (GPUs) to speed up such computations that ushered in new machine-learning and artificial-intelligence tools.

We come back to the prototypical model function $f(\mathbf{x};\theta)$ discussed in Chapter 2, and note that so far we have assumed that this function is computed on a classical computer. The classical computer is simply a physical device that can be controlled, where classical rules of discrete logic are run. In machine learning, more than such logic, we are interested in a



Expectation	$E(\theta, \mathbf{x}) = \langle\psi_\theta\|H(\mathbf{x})\|\psi_\theta\rangle$
VQA	$\theta^*(\mathbf{x}) = \arg\min_\theta E(\theta, \mathbf{x})$

Figure 5.1: Variational quantum algorithms use a quantum computer to generate variational quantum states and compute a function e.g., an expectation value. These expectation values can define a parameterized function that a classical computer can optimize, such as the energy of some quantum system. The optimization objective could be to minimize the energy. Implicit differentiation allows the computation of the gradient of the VQA solution without having to keep track of the classical optimization saving on the memory cost associated with keeping track of all intermediate steps in the VQA that reverse-mode automatic differentiation would require. In quantum problems, storing such intermediate steps in classical memory might be impossible.

continuous change of inputs to outputs. What if we replace this function with a quantum function run on quantum hardware $q(\mathbf{x}; \theta)$? We still have inputs that can be transformed by setting parameters θ, but now we have access to the Hilbert space of quantum states to perform transformations.

Quantum machine learning (QML) aims to use quantum hardware to process data and solve machine-learning problems. One of the promises of quantum machine learning is to be able to enhance the capabilities of machine-learning algorithms by exploiting the laws of quantum physics and tackling problems that are simply too difficult for classical computers to handle. In the rest of the chapter, we focus on a type of QML algorithm that is similar to classical machine learning — variational quantum algorithms (VQAs).

5.1 Variational quantum algorithms (VQAs)

A variational quantum algorithm computes a function $q(\mathbf{x}; \theta)$ on a quantum computer. The function could be part of a cost function defining a certain problem, e.g., finding the lowest energy state of a quantum system. If a quantum system's Hamiltonian is represented in a parameterized form $H(\mathbf{x})$, a VQA creates a parameterized state $|\psi(\mathbf{x}; \theta)\rangle$ using a quantum computer and measures expectation values used to construct an objective

function, such as energy,

$$\mathcal{L}(\theta) = q(\mathbf{x};\theta) = \langle\psi(\theta)|\, H(\mathbf{x})\, |\psi(\theta)\rangle\,. \tag{5.1}$$

A classical computer can optimize the variational parameters θ. The goal is to find a set of parameters that minimizes the cost function, leading to a solution of the problem, e.g., which configuration leads to the lowest energy for a system with Hamiltonian $H(\mathbf{x})$. The parameters can be concretely defined as a quantum circuit or unitary $U(\theta)$ that the quantum computer implements.

Variational quantum algorithms hold promise for several different tasks [39] such as computing ground-states, solving combinatorial optimization problems, error correction, and most relevant to this thesis — quantum machine-learning applications [38]. We can recognize that replacing a neural-network ansatz $f(\mathbf{x};\theta)$ with a quantum function $q(\mathbf{x};\theta)$ allows us to plug VQAs into all the problems discussed so far — discriminative modeling, generative models, or approximating unknown probability distributions.

So far, we have been trying to learn a classical model of this function in the form of a density matrix or a quantum process. In VQAs, we can directly compute the expectation value of any measurement observable on the quantum computer without requiring a model for the quantum system. We do not require simulations as the quantum computer emulates a quantum system.

In Paper IV, we are interested in enhancing the power of VQAs. Note that the computed solution to a problem, θ^*, will change as the problem is changed, e.g., if $\mathbf{x}$ represents the geometry of a molecule, changing the geometry defines a new Hamiltonian $H(\mathbf{x}')$, and therefore changes the solution. If we assume a solution function $\theta^* = z(\mathbf{x})$, the gradient $\partial_{\mathbf{x}} z(\mathbf{x})$ tells us how a change in the problem changes the solution. Let us denote this implicitly defined solution function as $\theta^*(\mathbf{x})$ to simplify the notation.

It is not always easy to have an analytical solution for optimization in a VQA. However, as we show and discuss in Paper IV, the gradient of the implicitly defined solution function can be computed by applying implicit differentiation. The requirement is having access to local gradients of some function $\partial_\theta h(\theta, \mathbf{x}|_{\theta^*})$ that defines a level-curve $h(\theta^*(\mathbf{x}), \mathbf{x}) = 0$. However, we must first discuss differentiation on a quantum computer before discussing implicit differentiation.

5.2 Computing gradients on a quantum computer

The function $q(\mathbf{x};\theta)$ can be written as the expectation value of some measurement operator $\mathcal{O}_\mathbf{x}$ on a state prepared by a unitary $U(\theta)$ starting from some initial state ρ_0. We can write the following general form for such a function:

$$q(\mathbf{x};\theta) = \mathrm{tr}\Big[\mathcal{O}_\mathbf{x} U(\theta)\rho_0 U^\dagger(\theta)\Big]. \tag{5.2}$$

The gradient of this function with respect to the components of the parameter θ can be computed by evaluating the function with two parameter settings $\theta^\pm$ using the so-called parameter shift rule [254, 255]

$$\frac{\partial q(\mathbf{x};\theta)}{\partial \theta_i} = \frac{1}{2\sin\alpha}\Big[q(\mathbf{x};\theta^+) - q(\mathbf{x};\theta^-)\Big]. \tag{5.3}$$

The parameter settings are $\theta^\pm = \theta \pm \alpha e_i$, with $e_i = 1$ for the i^{th} parameter and 0 otherwise for the component θ_i. The formula for the gradient is exact and analytical. An intuition can be developed by noticing that a unitary is parameterized by sine and cosine functions. The derivative of $\sin(\theta)$ is given by $\sin(\theta + \frac{\pi}{2}) - \sin(\theta - \frac{\pi}{2})$. Since we can only compute expectation values of the gradients, the parameter-shift rule provides an estimate that requires repeating measurements for each measurement operator. Further, the problem structure determines the total number of measurement settings necessary.

We can now replace our classical model function, e.g., a neural network $f(\mathbf{x};\theta)$, with the quantum model and apply gradient-based training to solve various problems. However, it was noted in Ref. [256] that random circuits, i.e., randomly initialized unitaries $U(\theta)$, suffer from the so-called barren-plateau (vanishing gradient) problem as the size of the quantum system increases. The barren-plateau problem refers to the fact that the variance of the gradient vanishes exponentially for random quantum circuits as the system size increases. Initializing a quantum circuit with a random parameter setting will therefore trap the optimization, giving no information on the direction of the steepest descent.

5.3 Implicit differentiation of VQAs

In 1638, René Descartes challenged the emerging mathematician Pierre de Fermat to determine the tangent to a complex curve known as the folium

of Descartes, given by the equation

$$x^3 + y^3 = 3axy. \tag{5.4}$$

Descartes was particularly fond of a method he developed to compute tangents. The folium of Descartes posed difficulties in finding tangents, except at the vertex, due to its cubic nature. Fermat, however, succeeded in finding tangents not only at the vertex but at any point on the curve. The method used by Fermat, implicit differentiation, is now common in calculus. Since the curve cannot be separated as a function of x only, by taking derivatives of the equation and rearranging terms, we can obtain an expression for the slope evaluated at any point:

$$\frac{dy}{dx} = \frac{-x^2 - ay}{y^2 - ax}. \tag{5.5}$$

The implicit nature comes from the assumption that there is a certain function $f(x, y) = 0$ that implicitly defines the curve $y(x)$ satisfying $f(x, y(x)) = 0$. We can differentiate f to obtain $\partial_x f, \partial_y f$ and use them to compute

$$\frac{dy}{dx} = -\frac{\partial_x f}{\partial_y f}|_{x,y}. \tag{5.6}$$

The implicit function theorem formally states the conditions that allow the calculation of implicit gradients. We refer to Paper IV for technical details, but present the theorem informally: if a function $h(\theta, \mathbf{x})$ is analytic and within a local neighborhood around $(\theta_0, \mathbf{x}_0)$ we have $h(\theta_0, \mathbf{x}_0) = 0$, then there exists an analytic solution function $\theta^*(\mathbf{x})$ that fulfills $h(\theta^*(\mathbf{x}), \mathbf{x}) = 0$. Since the solution function is analytic, its gradient $\partial_{\mathbf{x}}\theta^*(\mathbf{x})$ exists and can be obtained by implicit differentiation.

Implicit differentiation is playing a role in hyperparameter optimization, training neural ordinary differential equations, and has even led to the proposals of new architectures for machine learning such as deep equilibrium models [257]. Such advancements are possible due to the memory advantage provided by implicit differentiation over reverse-mode automatic differentiation (backpropagation) for gradient computation.

Implicit differentiation also helps in meta-optimization tasks. Consider an implicitly defined function as the solution to some optimization task

$$\theta^*(\mathbf{x}) = \arg\max_{\theta} g(\theta, \mathbf{x}). \tag{5.7}$$

The optimization could itself be carried out with gradient descent or any other method, such as convex optimization. If we want to compute the gradient $\partial_{\mathbf{x}}\theta^*(\mathbf{x})$, we can use implicit differentiation to avoid keeping track of all the intermediate steps and variables in the optimization that standard reverse-mode automatic differentiation (backpropagation) requires (incurring a huge memory cost). The optimality condition for the optimization can be simply

$$h(\theta, \mathbf{x}) = \partial_\theta g(\theta, \mathbf{x}) = 0. \tag{5.8}$$

Implicit differentiation can treat the inner optimization as a black box and provide the gradients $\partial_{\mathbf{x}}\theta^*(\mathbf{x})$ as long as we can find solutions for the optimality condition, and gradients $(\partial_\theta h, \partial_{\mathbf{x}} h)$ at the solution of the inner optimization. In meta-optimization tasks, where there is an inner optimization, e.g., hyperparameter optimization in machine learning, implicit differentiation computes hyper-gradients. Therefore it allows optimizing the hyperparameters guiding us towards a better choice of hyperparameters without having to go through a grid. Implicit differentiation has been used to optimize millions of hyperparameters in classical machine learning [258].

In most VQAs, we have an optimization objective and implicit differentiation can be applied to compute relevant physical quantities such as susceptibilities, and design algorithms that use gradients. In Ref. [259], implicit differentiation was used for inverse design and optimal control in open quantum systems, while Ref. [260] explored susceptibility calculations using implicit differentiation. We add to such applications with the hyperparameter optimization of QML algorithms and a new application — generation of entangled quantum states.

We now consider a variational quantum state $|\psi(\theta^*)\rangle$ that is generated as the solution of a VQA, e.g., after minimizing the expectation value $g(\theta, \mathbf{x}) = \langle\psi(\theta^*)| H(\mathbf{x}) |\psi(\theta^*)\rangle$ representing the energy of some parameterized Hamiltonian $H(x)$. At the solution point, the gradient $\partial_\theta g$ is zero, since it is a minimum (local). We can therefore define a function $f(\theta, \mathbf{x}) = \partial_\theta g = 0$ similar to the folium-of-Descartes problem. Applying implicit differentiation to this function, we get the gradient

$$\partial_x \theta = -[\partial_\theta g][\partial_x g]. \tag{5.9}$$

Therefore if we have access to the Hessian of the expectation value, i.e., $\partial_\theta^2 \langle\psi(\theta^*)| H(\mathbf{x}) |\psi(\theta^*)\rangle$, we can compute implicit gradients that tell us how the variational state will change as the Hamiltonian changes. The Hessian computation can be done with parameter-shift rules similar to the gradient

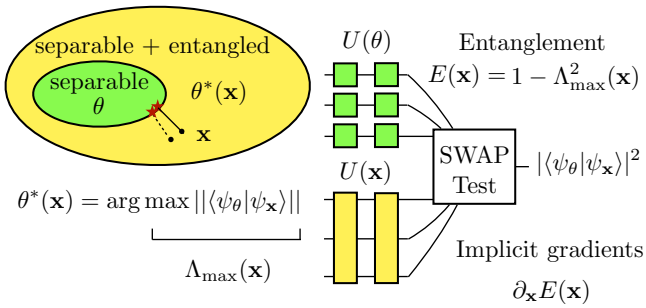


Figure 5.2: A geometric measure of entanglement defines the entanglement for a variational state $\psi_{\mathbf{x}}$ using an optimization over all possible separable states ψ_θ. This measure of entanglement $E(\mathbf{x})$, therefore, does not have an analytical form, so it is not possible to write the gradient $\partial_{\mathbf{x}} E(\mathbf{x})$ analytically. However, implicit differentiation can compute this gradient and can be used to optimize the parameters $\mathbf{x}$ to generate entangled states variationally as we show in Paper IV.

estimation. In case of problems with multiple parameters, the inverse term could be computationally expensive to calculate. However, using the idea of vector Jacobian products (VJPs), the inversion is efficiently approximated as solving a linear system of equations in an iterative manner. Paper IV contains more details and references for alternative techniques for this inversion, e.g., using a Neumann series expansion.

Implicit differentiation can therefore differentiate through an inner objective function, as we show in Paper IV. The gradients obtained therein allow computation of physically meaningful quantities such as susceptibilities or help in hyperparameter optimization of quantum machine-learning algorithms. We also demonstrate a new algorithm that creates entangled quantum states using gradients of a so-called geometric measure of entanglement and numerical optimization.

Now we can discuss the novel technique in Paper IV for entanglement generation. Our proposed technique can create entangled quantum states without ever explicitly defining what the state is, simply by optimizing an implicit definition of entanglement; see Fig. 5.2. The following inner optimization finds a separable state that has the highest overlap to another variational state $\psi_{\mathbf{x}}$:

$$\theta^* = \arg\max_{\theta} || \langle \psi_\theta | \psi_{\mathbf{x}} \rangle ||, \tag{5.10}$$

where ψ_θ is the set of all possible separable quantum states. The state

$\psi_{\mathbf{x}}$ is a general state that can be entangled. Using the optimized θ^*, we can define a geometric measure of entanglement $E(\mathbf{x})$ for the state that depends on the variational parameters $\mathbf{x}$, as we show in Fig. 5.2. The geometric measure does not have an analytical form in general, but we can compute the gradient $\partial_{\mathbf{x}} E(\mathbf{x})$ using implicit gradients. Now, we can optimize for maximum entanglement by tuning the variational parameters $\mathbf{x}$ using the gradient. In Paper IV, we show how such an approach generates a maximally entangled state.

We have not explored how such algorithms will perform in a quantum device where various types of noise degrades the calculation of the expectation values and gradients. We envisage a memory advantage that quantum hardware can provide for implicit gradient computation that could unlock the possibility of obtaining values of susceptibilities for large spin systems or generate large entangled states. The quantum computer can be used as a black box that gives access to solutions of VQAs, and local gradients.

Chapter 6

Summary of papers

Here we give an overview of the ten appended papers on which this thesis is based and connect the theoretical ideas discussed in the previous chapters to the results in those papers.

Paper I focuses on developing a method for quantum state tomography using conditional generative adversarial neural networks (QST-CGAN). The adversarial learning framework involves a generator and a discriminator neural network competing to learn an underlying probability distribution using data samples. Our approach utilizes a generator that gives probabilities for measurements on a quantum state by learning an underlying density matrix. We introduce a custom "DensityMatrix" layer that uses the Cholesky decomposition to ensure that the outputs from a neural network represent a valid quantum state, as discussed in Chapter 3. Several other generative modeling approaches are presented in Chapter 2 that could be explored to represent quantum states and learn the probability distribution of outcomes for measurements. While the "DensityMatrix" layer gives us an interpretable internal representation of the quantum state, it comes at the cost of poor scalability. However, in principle, a generative model can have any internal representation of the state and directly provide probabilities for measurement outcomes in a scalable way.

The "Expectation" layer in the generator implements the Born rule to obtain measurement probabilities. We show in Chapter 4 how the quantum-state-tomography problem is fundamentally an inversion problem that we solve by borrowing the idea of generative modeling with neural networks. Our approach can efficiently reconstruct an optical quantum state, requiring much fewer data points than iterative maximum likelihood estimation or

an accelerated projected-gradient method known as "superfast maximum likelihood". We also showcase the effectiveness of QST-CGAN on noisy experimental data by reconstructing a state from the measured Wigner function in an experiment. Finally, we show how such a technique can be adapted for the high-fidelity reconstruction of quantum states in a single shot after pretraining.

In Paper II, quantum state classification and reconstruction are connected to generative and discriminative modeling problems in machine learning. We demonstrate that a convolutional neural network can classify optical quantum states accurately, even in the presence of significant noise. The QST-CGAN approach, under different noise conditions, consistently achieves high reconstruction fidelities compared to alternative loss functions and iterative maximum likelihood estimation. Additionally, we investigate the effects of Gaussian convolution noise on reconstruction and demonstrate the successful reconstruction of mixed states with a small subset of data points, outperforming iterative maximum likelihood estimation. These results highlight the flexibility and effectiveness of the QST-CGAN approach for a wide range of scenarios, surpassing traditional methods.

Paper III is an experimental collaboration that created quantum states with negative Wigner-function values in a harmonic oscillator aided by a gradient-based optimization to estimate gate parameters. In this paper, the significant theoretical contribution was identifying a minimal set of gates that could be implemented to generate a wide variety of quantum states, such as a Gottesman-Kitaev-Preskill state and the first generation of the cubic phase state [261] in a 3D cavity. The two-step approach of finding optimal gates and then using pulse optimization to implement the gates was more robust than direct optimal control. The parameterization for the quantum operations in Paper III was fixed by the gates possible in the experiment. However, in Chapter 3, we discuss a more general way to parameterize unitary matrices that can be optimized to implement desired quantum operations.

In Paper IV, we draw on the knowledge about machine learning and automatic differentiation to show how implicit differentiation can be used in quantum physics and computing. The idea of implicit differentiation extended to quantum computers allows computing how ground-state properties of a variationally obtained quantum state, such as the ground-state energy, change as a function of the Hamiltonian parameters. Such ground-state gradients represent interesting physical quantities, such as generalized

susceptibilities in condensed matter systems, and this paper demonstrates how they can be computed automatically on a quantum computer. We also discuss how implicit differentiation allows the training of hyperparameters in a variational algorithm and a novel algorithm to create multipartite entangled quantum states enabled by gradient-based optimization of a geometric measure of entanglement. Automatic differentiation drives the success of current machine learning methods using neural networks. We discuss the basics of automatic differentiation and the implicit differentiation approach in Chapter 5.

In Paper V, we present a software tool for the classical simulation of quantum algorithms on quantum computers. Specifically, a new module was developed in QuTiP (Quantum Toolbox in Python), an open-source software for simulating quantum systems, enabling users to specify the parameters of a quantum computer and simulate quantum circuits running on that device at the pulse level.

In Paper VI, we transition from quantum state to quantum process tomography. Initially leveraging the architecture developed in Papers I and II, we discovered that neural networks were unnecessary for achieving accurate results. Instead, using gradient descent restricted to physically allowed quantum processes, combined with machine learning techniques such as batched training and regularization, proved highly effective in reconstructing quantum processes from measurement data. In several benchmarks, this gradient-descent quantum process tomography (GD-QPT) approach outperformed state-of-the-art techniques such as compressed sensing and projected least squares in handling sparse data and large system sizes simultaneously. A key to the success of the GD-QPT approach was using manifold optimization, discussed in Chapter 3, that implemented a trace-preserving condition on the Kraus operators.

Paper VII is about characterizing a new family of three-qubit quantum gates. In this paper, an experiment was carried out to implement these gates. The contribution to this work was solving the numerically challenging problem of three-qubit process tomography. We adapted a projected-least-squares technique augmented to include state-preparation and measurement errors from single-qubit gate set tomography. Implementing the process reconstruction algorithm using a combination of linear inversion and projection allowed fast computing process matrices, which made it possible to apply techniques such as bootstrap for uncertainty estimation.

In Paper VIII, the techniques developed in Paper VI were pivotal for the experimental characterization of a logical quantum gate in a bosonic error correction code. This experiment also used a gradient-based optimization to find the unitary operations that implemented a logical operation on an encoding. Since the GD-QPT method developed in Paper VI required very few data points, it was possible to perform a full process tomography experiment on a 32-dimensional Hilbert space. Previous methods for QPT would have required impractical amounts of measurement time to collect the necessary data for such a reconstruction. Similarly, as we discuss in Chapter 4, techniques such as compressed sensing would not be able to handle such a problem due to computational bottlenecks. The GD-QPT technique developed in Paper VI enabled this experiment that performed process tomography of a logical quantum operation with bosonic encoding without having to restrict to the two-dimensional logical subspace. Freed from this restriction, we could identify that the logical states were leaking out of the logical subspace.

In Paper IX, an experiment was performed to improve the readout of superconducting qubits using the higher energy levels of a transmon qubit. Improving the readout of such qubits is a crucial step towards quantum error correction. For this experiment, our contribution was developing a neural-network approach to help in the fast and accurate classification of the data representing different quantum states.

In Paper X, a Bayesian approach to quantum state tomography was adapted to reconstruct the quantum state of a photoelectron from noisy and sparse data. The photoelectron density matrix could be partially measured in an experiment using attosecond pulses but contained noise due to several factors, including a convolution of the signal with the response function of the measurement apparatus. The problem was further complicated by the limited data available for reconstructing the full density matrix. In this task, we took a Bayesian estimation approach that could include several priors based on physical motivations such as smoothness of the density matrix and spin-orbit coupling. As we discuss in Chapter 4, the Bayesian estimation made it possible to extract a meaningful representation of the quantum state along with uncertainty estimates. We could find the parameters describing the data using a complicated model that included the convolution operation using the Hamiltonian Monte Carlo method, as discussed in Chapter 3.

Chapter 7

Conclusion and outlook

In this thesis, we have discussed an interconnection between aspects of modeling, constrained optimization, and machine learning that applies to quantum physics problems. The pivotal contribution of this thesis lies in formulating a mindset for someone who would like to apply machine-learning methods to solve problems in quantum physics.

Machine learning, at its core, deals with data-driven predictive modeling. The Bayesian picture provides a general mindset to such problems, incorporating model building as the proposal of a likelihood function. Parameter-estimation techniques learn the model from data where constraints, priors, and regularization help in different ways.

We discuss the relationship between learning problems in quantum physics and machine learning. We showed that the central task of predicting measurement outcomes for a quantum system could be posed as a generative modeling task in machine learning since both relate to learning a model for an underlying data-generating process. Therefore, techniques developed to tackle challenges in generative modeling can be applied to quantum physics problems.

Generative models require approximations regarding the data-generating process we wish to learn to solve intractability issues. Neural networks are often employed in such approximations due to their universal function approximation guarantees. We can be similarly inspired in quantum learning tasks to use neural networks to approximate quantum states and operations. Gradient-based optimization that successfully trains neural networks can be very effective for quantum problems. We discussed how such gradient-based optimization applies to quantum state and process learning and estimating

parameters for target quantum operations.

Machine-learning approaches could tackle the problem of exponentially increasing Hilbert space dimension since they learn approximations from data. We could use machine-learning models as a proxy for quantum states and processes thereby finding effective models with a few parameters. Still, it is unclear how much machine-learning models, e.g., neural networks, approximate the true quantum system due to their black-box nature. An interesting future direction would be to use symbolic machine learning that finds efficient representations in an interpretable way. Also, using machine learning to reduce the complexity and cost of measurements in an experiment can be interesting to explore. Online learning of quantum systems and frequency re-calibration is possible with machine learning.

However, machine learning with gradient descent and neural networks is not the silver bullet to solve all problems in quantum physics. An intricate combination of suitable models that incorporate constraints and the structure of the problem is essential. Gradient descent and neural networks have been adapted to incorporate such structure and constraints, e.g., with manifold optimization for quantum process tomography. We can therefore tackle many learning problems in quantum physics by adapting gradient-based approaches with physics-informed models that effectively include constraints.

A significant and essential discussion in all such estimation problems is uncertainty computation. Neural-network-based models, or even a physical model, can only make predictions. However, the uncertainty associated with the predictions is essential in many experimental situations. Therefore we ended the discussion of learning problems with Bayesian estimation. However, this area is poorly explored compared to other ideas such as maximum likelihood estimation or projection-based learning. Bayesian estimation becomes computationally difficult for larger parameters. Exploring how to scale Bayesian estimation for quantum problems would be interesting.

Lastly, we discussed quantum machine learning — an emerging field. The implicit differentiation technique enhances the power of quantum machine learning and variational algorithms. However, our proposed ideas have only been tested in small simulations. It would be interesting to run actual calculations on a quantum computer to demonstrate implicit differentiation's benefits regarding memory cost. Since we only have access to the expectation value of quantities, it would become essential to analyze

the scaling or feasibility of the method in noisy quantum devices. Further studies and analysis would be required to address these questions and determine the practical applicability of the method.

Quantum mechanics and machine learning, therefore, represent two fascinating and non-intuitive fields with many avenues for exploration. This thesis hopes to spark inspiration for discoveries in both fields.

References

[1] A. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep Reinforcement Learning framework for Autonomous Driving", Electronic Imaging **2017**, 70 (2017).

[2] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection", in Proceedings of the 26th International Conference on Neural Information Processing Systems (2013), 2553–2561.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks", in Communications of the ACM, Vol. 60, edited by F Pereira, C. J. C. Burges, L Bottou, and K. Q. Weinberger, 6 (2017), pp. 84–90.

[4] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement", arXiv:1804.02767 (2018).

[5] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment Anything", arXiv:2304.02643 (2023).

[6] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis", in Proceedings of the 33rd international conference on machine learning, Vol. 48, edited by M. F. Balcan and K. Q. Weinberger, Proceedings of Machine Learning Research (2016), pp. 1060–1069.

[7] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks", in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017), pp. 5967–5976.

[8] B. Li, X. Qi, T. Lukasiewicz, and P. Torr, "Controllable text-to-image generation", in Advances in Neural Information Processing Systems, Vol. 32 (2019).

[9] P. Dhariwal and A. Nichol, "Diffusion Models Beat GANs on Image Synthesis", Advances in Neural Information Processing Systems **11**, 8780 (2021).

[10] 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, edited by I Guyon, U. V. Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett (2017), pp. 5999–6009.

[11] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "LLaMA: Open and Efficient Foundation Language Models", arXiv:2302.13971 (2023).

[12] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro, and Y. Zhang, "Sparks of Artificial General Intelligence: Early experiments with GPT-4", arXiv:2303.12712 (2023).

[13] J. Jumper et al., "Highly accurate protein structure prediction with AlphaFold", Nature **596**, 583 (2021).

[14] J. Kirkpatrick et al., "Pushing the frontiers of density functionals by solving the fractional electron problem", Science **374**, 1385 (2021).

[15] J. Wang and L. Hu, "Solving Low-Rank Semidefinite Programs via Manifold Optimization", arXiv:2303.01722, 1 (2023).

[16] J. L. Watson et al., "De novo design of protein structure and function with RFdiffusion.", Nature, 10.1038/s41586-023-06415-8 (2023).

[17] S. M. Udrescu and M. Tegmark, "AI Feynman: A physics-inspired method for symbolic regression", Science Advances **6**, eaay2631 (2020).

[18] R. Guimerà, I. Reichardt, A. Aguilar-Mogas, F. A. Massucci, M. Miranda, J. Pallarès, and M. Sales-Pardo, "A Bayesian machine scientist to aid in the solution of challenging scientific problems", Science Advances **6**, eaav6971 (2020).

[19] M. Krenn, R. Pollice, S. Y. Guo, M. Aldeghi, A. Cervera-Lierta, P. Friederich, G. dos Passos Gomes, F. Häse, A. Jinich, A. Nigam, Z. Yao, and A. Aspuru-Guzik, "On scientific understanding with artificial intelligence", Nature Reviews Physics **4**, 761 (2022).

[20] A. Fawzi, M. Balog, A. Huang, T. Hubert, B. Romera-Paredes, M. Barekatain, A. Novikov, F. J. Francisco, J. Schrittwieser, G. Swirszcz, D. Silver, D. Hassabis, and P. Kohli, "Discovering faster matrix multiplication algorithms with reinforcement learning", Nature **610**, 47 (2022).

[21] E. Real, C. Liang, D. So, and Q. Le, "AutoML-zero: evolving machine learning algorithms from scratch", in Proceedings of the 37th international conference on machine learning, Vol. 119, edited by H. D. III and A. Singh, Proceedings of Machine Learning Research (2020), pp. 8007–8019.

[22] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play", Science **362**, 1140 (2018).

[23] T. N. Theis and H. S. Philip Wong, "The End of Moore's Law: A New Beginning for Information Technology", Computing in Science and Engineering **19**, 41 (2017).

[24] D. S. Abrams and S. Lloyd, "Simulation of many-body fermi systems on a universal quantum computer", Physical Review Letters **79**, 2586 (1997).

[25] J. D. Whitfield, P. J. Love, and A. Aspuru-Guzik, "Computational complexity in electronic structure", Physical Chemistry Chemical Physics **15**, 397 (2013).

[26] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love, and M. Head-Gordon, "Simulated Quantum Computation of Molecular Energies", Science **309**, 1704 (2005).

[27] Y. Manin, *Computable and Uncomputable (in Russian)*, Sovetskoye Radio, Moscow, 1980.

[28] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines", Journal of Statistical Physics **22**, 563 (1980).

[29] R. P. Feynman, "Simulating physics with computers", International Journal of Theoretical Physics **21**, 467 (1982).

[30] D. R. Simon, "On the power of quantum computation", SIAM journal on computing **26**, 1474 (1997).

[31] U. Vazirani, "On the power of quantum computation", Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **356**, 1759 (1998).

[32] C. H. Bennett and D. P. DiVincenzo, "Quantum information and computation", Nature **404**, 247 (2000).

[33] F. Arute et al., "Quantum supremacy using a programmable superconducting processor", Nature **574**, 505 (2019).

[34] I. H. Deutsch, "Harnessing the Power of the Second Quantum Revolution", PRX Quantum **1**, 20101 (2020).

[35] H. S. Zhong et al., "Quantum computational advantage using photons", Science **370**, 1460 (2020).

[36] A. J. Daley, I. Bloch, C. Kokail, S. Flannigan, N. Pearson, M. Troyer, and P. Zoller, "Practical quantum advantage in quantum simulation", Nature **607**, 667 (2022).

[37] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. van den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme, and A. Kandala, "Evidence for the utility of quantum computing before fault tolerance", Nature **618**, 500 (2023).

[38] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning", Nature **549**, 195 (2017).

[39] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, "Variational quantum algorithms", Nature Reviews Physics **3**, 625 (2021).

[40] M. Schuld and N. Killoran, "Is quantum advantage the right goal for quantum machine learning?", PRX Quantum **3**, 030101 (2022).

[41] G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks", Science **355**, 602 (2017).

[42] V. Gebhart, R. Santagati, A. A. Gentile, E. M. Gauger, D. Craig, N. Ares, L. Banchi, F. Marquardt, L. Pezzè, and C. Bonato, "Learning quantum systems", Nature Reviews Physics **5**, 141 (2023).

[43] A. Pais, "Einstein and the quantum theory", Reviews of Modern Physics **51**, 863 (1979).

[44] A. Sudbery, *Quantum Mechanics and the Particles of Nature: An Outline for Mathematicians* (Cambridge University Press, 1986).

[45] R. F. Bader, "A Quantum Theory of Molecular Structure and Its Applications", Chemical Reviews **91**, 893 (1991).

[46] C. C. Gerry and P. L. Night, *Introductory Quantum Optics* (Cambridge University Press, 2005).

[47] D. Harlow, "Jerusalem lectures on black holes and quantum information", Reviews of Modern Physics **88**, 15002 (2016).

[48] M. J. Klein, "Max Planck and the beginnings of the quantum theory", Archive for History of Exact Sciences **1**, 459 (1975).

[49] M. Planck, "On the Law of the Energy Distribution in the Normal Spectrum", Annalen der Physik **4**, 553 (1901).

[50] A. I. Sabra and A. Zajac, "Theories of Light, from Descartes to Newton", American Journal of Physics **52**, 862 (1984).

[51] T. Young, "II. The Bakerian Lecture. On the theory of light and colours", Philosophical Transactions of the Royal Society of London **92**, 12 (1802).

[52] N. Forbes and B. Mahon, *Faraday, maxwell, and the electromagnetic field: how two men revolutionized physics* (Prometheus Books, 2014).

[53] J. C. Maxwell, "A Dynamical Theory of the Electromagnetic Field", Nature **119**, 125 (1927).

[54] A. Einstein, "Über einen die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichtspunkt", Annalen der Physik **322**, 132 (1905).

[55] A. H. Compton, "A Quantum Theory of the Scattering of X-rays by Light Elements", Physical Review **21**, 483 (1923).

[56] L. Gurvits, "Classical deterministic complexity of Edmonds' Problem and quantum entanglement", in Proceedings of the thirty-fifth annual acm symposium on theory of computing (2003), pp. 10–19.

[57] S. Gharibian, "Strong NP-hardness of the quantum separability problem", Quantum Information and Computation **10**, 343 (2010).

[58] A. Shimony, "Degree of Entanglement", Annals of the New York Academy of Sciences **755**, 675 (1995).

[59] T.-C. Wei and P. M. Goldbart, "Geometric measure of entanglement and applications to bipartite and multipartite quantum states", Physical Review A **68**, 042307 (2003).

[60] A. Einstein, B. Podolsky, and N. Rosen, "Can quantum-mechanical description of physical reality be considered complete?", Physical Review **47**, 777 (1935).

[61] J. S. Bell, "On the einstein podolsky rosen paradox", Physics Physique Fizika **1**, 195 (1964).

[62] S. J. Freedman and J. F. Clauser, "Experimental test of local hidden-variable theories", Phys. Rev. Lett. **28**, 938 (1972).

[63] A. Aspect, P. Grangier, and G. Roger, "Experimental tests of realistic local theories via bell's theorem", Phys. Rev. Lett. **47**, 460 (1981).

[64] A. Aspect, J. Dalibard, and G. Roger, "Experimental test of bell's inequalities using time-varying analyzers", Phys. Rev. Lett. **49**, 1804 (1982).

[65] R. Orús, "Tensor networks for complex quantum systems", Nature Reviews Physics **1**, 538 (2019).

[66] I. A. Luchnikov, S. V. Vintskevich, H. Ouerdane, and S. N. Filippov, "Simulation Complexity of Open Quantum Dynamics: Connection with Tensor Networks", Physical Review Letters **122**, 160401 (2019).

[67] H. S. Zhong et al., "Phase-Programmable Gaussian Boson Sampling Using Stimulated Squeezed Light", Physical Review Letters **127**, 180502 (2021).

[68] Y. Wu et al., "Strong Quantum Computational Advantage Using a Superconducting Quantum Processor", Physical Review Letters **127**, 180501 (2021).

[69] L. S. Madsen et al., "Quantum computational advantage with a programmable photonic processor", Nature **606**, 75 (2022).

[70] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2010).

[71] D. Gross, Y. K. Liu, S. T. Flammia, S. Becker, and J. Eisert, "Quantum state tomography via compressed sensing", Physical Review Letters **105**, 1 (2010).

[72] B. Qi, Z. Hou, L. Li, D. Dong, G. Xiang, and G. Guo, "Quantum State Tomography via Linear Regression Estimation", Scientific Reports **3**, 3496 (2013).

[73] A. V. Rodionov, A. Veitia, R. Barends, J. Kelly, D. Sank, J. Wenner, J. M. Martinis, R. L. Kosut, and A. N. Korotkov, "Compressed sensing quantum process tomography for superconducting quantum gates", Physical Review B **90**, 144504 (2014).

[74] E. Prugovečki, "Information-theoretical aspects of quantum measurement", International Journal of Theoretical Physics **16**, 321 (1977).

[75] G. M. D'Ariano, D. F. Magnani, and P. Perinotti, "Adaptive Bayesian and frequentist data processing for quantum tomography", Physics Letters A **373**, 1111 (2009).

[76] C. H. Baldwin, I. H. Deutsch, and A. Kalev, "Strictly-complete measurements for bounded-rank quantum-state tomography", Physical Review A **93**, 052105 (2016).

[77] A. Czerwinski, "Quantum state tomography with informationally complete POVMs generated in the time domain", Quantum Information Processing **20**, 105 (2021).

[78] Y. S. Teo, S. Shin, H. Jeong, Y. Kim, Y. H. Kim, G. I. Struchalin, E. V. Kovlakov, S. S. Straupe, S. P. Kulik, G. Leuchs, and L. L. Sánchez-Soto, "Benchmarking quantum tomography completeness and fidelity with machine learning", New Journal of Physics **23** (2021).

[79] S. T. Flammia, D. Gross, Y.-K. Liu, and J. Eisert, "Quantum tomography via compressed sensing: error bounds, sample complexity and efficient estimators", New Journal of Physics **14**, 095022 (2012).

[80] R. O'Donnell and J. Wright, "Quantum spectrum testing", in Proceedings of the forty-seventh annual acm symposium on theory of computing, STOC '15 (2015), 529–538.

[81] R. Kueng, H. Rauhut, and U. Terstiege, "Low rank matrix recovery from rank one measurements", Applied and Computational Harmonic Analysis **42**, 88 (2017).

[82] R. O'Donnell and J. Wright, "Efficient quantum tomography", in Proceedings of the forty-eighth annual ACM symposium on Theory of Computing (2016), pp. 899–912.

[83] C. Shen, R. W. Heeres, P. Reinhold, L. Jiang, Y. K. Liu, R. J. Schoelkopf, and L. Jiang, "Optimized tomography of continuous variable systems using excitation counting", Physical Review A **94**, 052327 (2016).

[84] D. Sych, J. Řeháček, Z. Hradil, G. Leuchs, and L. L. Sánchez-Soto, "Informational completeness of continuous-variable measurements", Physical Review A **86**, 052123 (2012).

[85] J. M. Renes, R. Blume-Kohout, A. J. Scott, and C. M. Caves, "Symmetric informationally complete quantum measurements", Journal of Mathematical Physics **45**, 2171 (2004).

[86] C. Fuchs, M. Hoang, and B. Stacey, "The SIC Question: History and State of Play", Axioms **6**, 21 (2017).

[87] H. Zhu, "Quantum state estimation with informationally overcomplete measurements", Physical Review A **90**, 012115 (2014).

[88] V. V. Albert, K. Noh, K. Duivenvoorden, D. J. Young, R. T. Brierley, P. Reinhold, C. Vuillot, L. Li, C. Shen, S. M. Girvin, B. M. Terhal, and L. Jiang, "Performance and structure of single-mode bosonic codes", Physical Review A **97**, 032346 (2018).

[89] A. L. Grimsmo, J. Combes, and B. Q. Baragiola, "Quantum Computing with Rotation-Symmetric Bosonic Codes", Physical Review X **10**, 11058 (2020).

[90] J. Carrasquilla and R. G. Melko, "Machine learning phases of matter", Nature Physics **13**, 431 (2017).

[91] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", in Proceedings of the 2001 ieee computer society conference on computer vision and pattern recognition. cvpr 2001, Vol. 1 (IEEE, 2001), pp. 511–518.

[92] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: real and stealthy attacks on state-of-the-art face recognition", in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16 (2016), 1528–1540.

[93] T. B. Brown et al., "Language Models are Few-Shot Learners", arXiv:2005.14165 (2020).

[94] D. Silver et al., "Mastering the game of go without human knowledge", Nature **550**, 354 (2017).

[95] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, "Mastering Atari, Go, chess and shogi by planning with a learned model", Nature **588**, 604 (2020).

[96] L. Gatys, A. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style", Journal of Vision **16**, 326 (2016).

[97] G. Branwen, *GPT-3 Creative Fiction*, Accessed on 10.03.2021, (2020) https://www.gwern.net/GPT-3.

[98] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, D. Guo, and C. Blundell, "Agent57: Outperforming the atari human benchmark", (2020).

[99] J. Jumper et al., "High Accuracy Protein Structure Prediction Using Deep Learning", in Fourteenth Critical Assessment of Techniques for Protein Structure Prediction (Abstract Book) (2020), pp. 22–24.

[100] N. Wittler, F. Roy, K. Pack, M. Werninghaus, A. S. Roy, D. J. Egger, S. Filipp, F. K. Wilhelm, and S. Machnes, "Integrated tool set for control, calibration, and characterization of quantum devices applied to superconducting qubits", Phys. Rev. Appl. **15**, 034080 (2021).

[101] M. Krenn, M. Erhard, and A. Zeilinger, "Computer-inspired quantum experiments", Nature Reviews Physics **2**, 649 (2020).

[102] B. Lienhard et al., "Deep-neural-network discrimination of multiplexed superconducting-qubit states", Phys. Rev. Appl. **17**, 014024 (2022).

[103] P. Duan, Z.-F. Chen, Q. Zhou, W.-C. Kong, H.-F. Zhang, and G.-P. Guo, "Mitigating Crosstalk-Induced Qubit Readout Error with Shallow-Neural-Network Discrimination", Physical Review Applied **16**, 024063 (2021).

[104] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge", International Journal of Computer Vision **115**, 211 (2015).

[105] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep Unsupervised Learning using Nonequilibrium Thermodynamics", 32nd International Conference on Machine Learning, ICML 2015 **3**, 2246 (2015).

[106] I. Bengtsson and K. Życzkowski, *Geometry of quantum states: An introduction to quantum entanglement* (Cambridge University Press, 2006).

[107] G. Torlai, G. Mazzola, G. Carleo, and A. Mezzacapo, "Precise measurement of quantum observables with neural-network estimators", Physical Review Research **2**, 022060 (2020).

[108] G. Torlai, B. Timar, E. P. 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", Physical Review Letters **123**, 19 (2019).

[109] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, "Reconstructing quantum states with generative models", Nature Machine Intelligence **1**, 155 (2019).

[110] A. M. Palmieri, E. Kovlakov, F. Bianchi, D. Yudin, S. Straupe, J. D. Biamonte, and S. Kulik, "Experimental neural network enhanced quantum tomography", npj Quantum Information **6**, 20 (2020).

[111] D. Luo, Z. Chen, J. Carrasquilla, and B. K. Clark, "Autoregressive Neural Network for Simulating Open Quantum Systems via a Probabilistic Formulation", Physical Review Letters **128**, 090501 (2022).

[112] "Diffusion-GAN: Training GANs with Diffusion", arXiv:2206.02262 (2022).

[113] K. Pandey, A. Mukherjee, P. Rai, and A. Kumar, "DiffuseVAE: Efficient, Controllable and High-Fidelity Generation from Low-Dimensional Latents", arXiv:2201.00308 (2022).

[114] A. Sauer, K. Schwarz, and A. Geiger, "StyleGAN-XL: Scaling StyleGAN to Large Diverse Datasets", in Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings (2022), pp. 1–10.

[115] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", SIAM Journal on Computing **26**, 1484 (1997).

[116] S. Aaronson, "Guest Column: NP-complete problems and physical reality", ACM SIGACT News **36**, 30 (2005).

[117] C. P. Koch, U. Boscain, T. Calarco, G. Dirr, S. Filipp, S. J. Glaser, R. Kosloff, S. Montangero, T. Schulte-Herbrüggen, D. Sugny, and F. K. Wilhelm, "Quantum optimal control in quantum technologies. Strategic report on current status, visions and goals for research in Europe", EPJ Quantum Technology **9**, 19 (2022).

[118] M. Y. Niu, S. Boixo, V. N. Smelyanskiy, and H. Neven, "Universal quantum control through deep reinforcement learning", npj Quantum Information **5**, 33 (2019).

[119] V. V. Sivak, A. Eickbusch, H. Liu, B. Royer, I. Tsioutsios, and M. H. Devoret, "Model-Free Quantum Control with Reinforcement Learning", Physical Review X **12**, 011059 (2022).

[120] H. Ball, M. J. Biercuk, A. R. R. Carvalho, J. Chen, M. Hush, L. A. De Castro, L. Li, P. J. Liebermann, H. J. Slatyer, C. Edmunds, V. Frey, C. Hempel, and A. Milne, "Software tools for quantum control: improving quantum computer performance through noise and error suppression", Quantum Science and Technology **6**, 044011 (2021).

[121] A. Montanaro, "Quantum algorithms: an overview", npj Quantum Information **2**, 15023 (2016).

[122] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, "Noisy intermediate-scale quantum algorithms", Reviews of Modern Physics **94**, 015004 (2022).

[123] J. R. Brink, "John von neumann reconsidered", IEEE Annals of the History of Computing **11**, 179–181 (1989).

[124] A. M. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem. A Correction", Proceedings of the London Mathematical Society **s2-43**, 544 (1938).

[125] D. K. Allison, "Papers of John von Neumann on Computing and Computer Theory . John von Neumann , William Aspray , Arthur Burks", Isis **78**, 603 (1987).

[126] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms* (MIT press, 2022).

[127] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", Communications of the ACM **60**, 84–90 (2017).

[128] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention", 32nd International Conference on Machine Learning, ICML 2015 **3**, 2048 (2015).

[129] Alan M Turing, *Intelligent machinery*, Accessed on 10.03.2021, (1948) http://www.alanturing.net/intelligent_machinery/.

[130] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects", Science **349**, 255 (2015).

[131] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", Nature **521**, 436 (2015).

[132] R. M. Neal, "Pattern recognition and machine learning", Technometrics **49**, 366 (2007).

[133] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning* (Cambridge University Press, 2014).

[134] H. W. Lin, M. Tegmark, and D. Rolnick, "Why Does Deep and Cheap Learning Work So Well?", Journal of Statistical Physics **168**, 1223 (2017).

[135] K. P. Murphy, *Machine learning: a probabilistic perspective* (MIT press, 2012).

[136] J. Gasteiger and J. Zupan, "Neural Networks in Chemistry", Angewandte Chemie International Edition in English **32**, 503 (1993).

[137] W. Duch and G. H. Diercksen, "Neural networks as tools to solve problems in physics and chemistry", Computer Physics Communications **82**, 91 (1994).

[138] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli, "Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next", Journal of Scientific Computing **92**, 88 (2022).

[139] P. Reiser, M. Neubert, A. Eberhard, L. Torresi, C. Zhou, C. Shao, H. Metni, C. van Hoesel, H. Schopmans, T. Sommer, and P. Friederich, "Graph neural networks for materials science and chemistry", Communications Materials **3**, 93 (2022).

[140] J. Jiménez-Luna, F. Grisoni, and G. Schneider, "Drug discovery with explainable artificial intelligence", Nature Machine Intelligence **2**, 573 (2020).

[141] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, "Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges", arXiv:2104.13478 (2021).

[142] A. Karpatne, G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar, "Theory-Guided Data Science: A New Paradigm for Scientific Discovery from Data", IEEE Transactions on Knowledge and Data Engineering **29**, 2318 (2017).

[143] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations", Journal of Computational Physics **378**, 686 (2019).

[144] G. Nakamura, *Inverse Modeling An introduction to the theory and methods of inverse problems and data assimilation* (IOP Publishing, 2015).

[145] G. Bal, "Introduction to inverse problems", Lecture Notes - Department of Applied Physics and Applied Mathematics, Columbia University, New York (2012).

[146] M. Burger and H. W. Engl, "Training neural networks with noisy data as an ill-posed problem", Advances in Computational Mathematics **13**, 335 (2000).

[147] A. L. Blum and R. L. Rivest, "Training a 3-node neural network is NP-complete", in *Proceedings of the 1st Annual Workshop on Computational Learning Theory, COLT 1988* (1993), pp. 9–28.

[148] V. Froese and C. Hertrich, "Training Neural Networks is NP-Hard in Fixed Dimension", arXiv:2303.17045 (2023).

[149] A. D. Kulkarni, "Solving ill-posed problems with artificial neural networks", Neural Networks **4**, 477 (1991).

[150] J. Adler and O. Öktem, "Solving ill-posed inverse problems using iterative deep neural networks", Inverse Problems **33**, 124007 (2017).

[151] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks", in IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 43, 12 (2021), pp. 4217–4228.

[152] P. Wang, *This person does not exist*, Accessed on 10.08.2023, (2019) `https://thispersondoesnotexist.com`.

[153] G. Cybenko, "Approximation by superpositions of a sigmoidal function", Mathematics of control, signals and systems **2**, 303 (1989).

[154] N. Cotter, "The Stone-Weierstrass theorem and its application to neural networks", IEEE Transactions on Neural Networks **1**, 290 (1990).

[155] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators", Neural Networks **2**, 359 (1989).

[156] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function", Neural Networks **6**, 861 (1993).

[157] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning* (MIT press, 2016).

[158] S. Moon, "ReLU Network with Bounded Width Is a Universal Approximator in View of an Approximate Identity", Applied Sciences **11**, 427 (2021).

[159] O. Delalleau and Y. Bengio, "Shallow vs. deep sum-product networks", in Advances in Neural Information Processing Systems, Vol. 24, edited by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger (2011).

[160] H. N. Mhaskar and T. Poggio, "Deep vs. shallow networks: An approximation theory perspective", Analysis and Applications **14**, 829 (2016).

[161] R. Eldan and O. Shamir, "The power of depth for feedforward neural networks", in 29th annual conference on learning theory, Vol. 49, edited by V. Feldman, A. Rakhlin, and O. Shamir, Proceedings of Machine Learning Research (2016), pp. 907–940.

[162] N. Cohen, O. Sharir, and A. Shashua, "On the expressive power of deep learning: a tensor analysis", in 29th Annual Conference on Learning Theory, Vol. 49, Proceedings of Machine Learning Research (2016), pp. 698–728.

[163] K. Malcolm and J. Casco-Rodriguez, "A Comprehensive Review of Spiking Neural Networks: Interpretation, Optimization, Efficiency, and Best Practices", arXiv:2303.10780 (2023).

[164] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors", Nature **323**, 533 (1986).

[165] G. E. Hinton, "A practical guide to training restricted boltzmann machines", in *Neural networks: tricks of the trade: second edition*, edited by G. Montavon, G. B. Orr, and K.-R. Müller (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012), pp. 599–619.

[166] G. E. Hinton, "Training Products of Experts by Minimizing Contrastive Divergence", Neural Computation **14**, 1771 (2002).

[167] A. Krizhevsky, *Learning multiple layers of features from tiny images*, tech. rep. (University of Toronto, 2009).

[168] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes", 2nd International Conference on Learning Representations, ICLR (2014).

[169] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets", in Advances in Neural Information Processing Systems, Vol. 27 (2014).

[170] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets", arXiv:1411.1784 (2014).

[171] D. Rezende and S. Mohamed, "Variational Inference with Normalizing Flows", in Proceedings of the 32nd International Conference on Machine Learning (2015), pp. 1530–1538.

[172] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-Based Generative Modeling Through Stochastic Differential Equations", 9th International Conference on Learning Representations, ICLR (2021).

[173] D. P. Kingma and M. Welling, "An Introduction to Variational Autoencoders", Foundations and Trends® in Machine Learning **12**, 307 (2019).

[174] A. M. Andrew, *Information Theory, Inference, and Learning Algorithms*, Vol. 33 (Cambridge university press, 2004).

[175] M. Betancourt, "A Conceptual Introduction to Hamiltonian Monte Carlo", arXiv:1701.02434 (2017).

[176] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially Learned Inference", in 5th International Conference on Learning Representations, ICLR (2017).

[177] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks", in 4th international conference on learning representations, ICLR 2016, san juan, puerto rico, may 2-4, 2016, conference track proceedings, edited by Y. Bengio and Y. LeCun (2016).

[178] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric", in Proceedings of The 33rd International Conference on Machine Learning, Vol. 48, edited by M. F. Balcan and K. Q. Weinberger, Proceedings of Machine Learning Research (2016), pp. 1558–1566.

[179] S. Arora and Y. Zhang, "Do GANs actually learn the distribution? An empirical study", arXiv:1706.08224 (2017).

[180] A. Hyvärinen, "Estimation of non-normalized statistical models by score matching", Journal of Machine Learning Research **6**, 695 (2005).

[181] Y. Song, S. Garg, J. Shi, and S. Ermon, "Sliced Score Matching: A Scalable Approach to Density and Score Estimation", Proceedings of Machine Learning Research **115**, 574 (2019).

[182] G. Parisi, "Correlation functions and computer simulations", Nuclear Physics B **180**, 378 (1981).

[183] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, W. Zhang, B. Cui, and M.-H. Yang, "Diffusion Models: A Comprehensive Survey of Methods and Applications", arXiv:2209.00796 **1** (2022).

[184] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations", in Advances in neural information processing systems 31, edited by S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (2018), pp. 6572–6583.

[185] S. I. Kabanikhin, "Definitions and examples of inverse and ill-posed problems", Journal of Inverse and Ill-Posed Problems **16**, 317 (2008).

[186] A. Lucas, M. Iliadis, R. Molina, and A. K. Katsaggelos, "Using Deep Neural Networks for Inverse Problems in Imaging: Beyond Analytical Methods", IEEE Signal Processing Magazine **35**, 20 (2018).

[187] E. De Vito, L. Rosasco, A. Caponnetto, U. De Giovannini, F. Odone, and P. Bartlett, "Learning from examples as an inverse problem.", Journal of Machine Learning Research **6** (2005).

[188] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning", Annals of Statistics **36**, 1171 (2008).

[189] A. R. Syversveen, "Noninformative Bayesian priors. Interpretation and problems with construction and applications", Preprint statistics **3**, 1 (1998).

[190] F. G. Waqar, S. Patel, and C. M. Simon, "A tutorial on the Bayesian statistical approach to inverse problems", arXiv:2304.07610 (2023).

[191] S. M. Stigler, "The Epic Story of Maximum Likelihood", Statistical Science **22**, 598 (2007).

[192] J. R. Magnus, "Gauss on least-squares and maximum-likelihood estimation", Archive for History of Exact Sciences **76**, 425 (2022).

[193] V. Terekhovich, "Metaphysics of the principle of least action", Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics **62**, 189 (2018).

[194] *"Gradient descent can write code better than you"*, `https://twitter.com/karpathy/status/893576281375219712`.

[195] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro, "Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism", arXiv:1909.08053 (2019).

[196] A. Sriram, A. Das, B. M. Wood, S. Goyal, and C. L. Zitnick, "Towards training billion parameter graph neural networks for atomic simulations", (2022).

[197] J. Fearnley, P. Goldberg, A. Hollender, and R. Savani, "The Complexity of Gradient Descent: CLS = PPAD PLS", Journal of the ACM **70**, 1 (2023).

[198] R. Tibshirani, "Regression Shrinkage and Selection Via the Lasso", Journal of the Royal Statistical Society: Series B (Methodological) **58**, 267 (1996).

[199] L. Freijeiro-González, M. Febrero-Bande, and W. González-Manteiga, "A Critical Review of LASSO and Its Derivatives for Variable Selection Under Dependence Among Covariates", International Statistical Review **90**, 118 (2022).

[200] T. A. N., "On the solution of improperly posed problems and the method of regularization", in Sov. Math. Vol. 5, 3 (Russian Academy of Sciences, 1963), p. 1035.

[201] D. N. Schreiber-Gregory, "Ridge Regression and multicollinearity: An in-depth review", Model Assisted Statistics and Applications **13**, edited by J. Waller and T. Smith, 359 (2018).

[202] G. W. S. and D. S. Watkins, *Fundamentals of Matrix Computations.* Vol. 59, 199 (John Wiley & Sons, 1992), p. 299.

[203] K. Banaszek, G. M. D'Ariano, M. G. A. Paris, and M. F. Sacchi, "Maximum-likelihood estimation of the density matrix", Physical Review A **61**, 010304 (1999).

[204] L. Jing, Y. Shen, T. Dubček, J. Peurifoy, S. Skirlo, Y. LeCun, M. Tegmark, and M. Soljačić, "Tunable Efficient Unitary Neural Networks (EUNN) and their application to RNNs", 34th International Conference on Machine Learning, ICML 2017 **4**, 2753 (2016).

[205] R. W. Heeres, B. Vlastakis, E. Holland, S. Krastanov, V. V. Albert, L. Frunzio, L. Jiang, and R. J. Schoelkopf, "Cavity State Manipulation Using Photon-Number Selective Phase Gates", Physical Review Letters **115**, 137002 (2015).

[206] B. T. Kiani, S. Lloyd, and R. Maity, "Learning Unitaries by Gradient Descent", arXiv:2001.11897 (2020).

[207] B. Kiani, R. Balestriero, Y. LeCun, and S. Lloyd, "projUNN: efficient method for training deep networks with unitary matrices", arXiv:2203.05483, 1 (2022).

[208] C. J. Wood, J. D. Biamonte, and D. G. Cory, "Tensor networks and graphical calculus for open quantum systems", Quantum Information and Computation **15**, 759 (2011).

[209] M.-D. Choi, "Completely positive linear maps on complex matrices", Linear Algebra and its Applications **10**, 285 (1975).

[210] S. Srinivasan, S. Adhikary, J. Miller, G. Rabusseau, and B. Boots, "Towards a Trace-Preserving Tensor Network Representation of Quantum Channels", Second Workshop on Quantum Tensor Networks in Machine Learning In conjunction with NeurIPS (2021).

[211] P. Kidger, "On Neural Differential Equations", arXiv:2202.02435 (2022).

[212] S. Ruder, "An overview of gradient descent optimization algorithms", arXiv:1609.04747 (2016).

[213] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization", 3rd International Conference on Learning Representations, ICLR, 1 (2015).

[214] M. A. Nielsen, *Neural Networks and Deep Learning* (Determination Press, 2015).

[215] A Griewank, "On automatic differentiation. Preprint ANL/MCS-P10-1088", Mathematical Programming: Recent Developments and Applications **6**, 83 (1989).

[216] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey", Journal of Machine Learning Research **18**, 1 (2018).

[217] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds* (Princeton University Press, 2008).

[218] N. Boumal, *An Introduction to Optimization on Smooth Manifolds* (Cambridge University Press, 2023).

[219] W. Ring and B. Wirth, "Optimization Methods on Riemannian Manifolds and Their Application to Shape Space", SIAM Journal on Optimization **22**, 596 (2012).

[220] T. Homem-de Mello and G. Bayraksan, "Monte Carlo sampling-based methods for stochastic optimization", Surveys in Operations Research and Management Science **19**, 56 (2014).

[221] S. Duane, A. Kennedy, B. J. Pendleton, and D. Roweth, "Hybrid Monte Carlo", Physics Letters B **195**, 216 (1987).

[222] M. D. Hoffman and A. Gelman, "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo", Journal of Machine Learning Research **15**, 1593 (2011).

[223] E. Candès and B. Recht, "Exact matrix completion via convex optimization", Communications of the ACM **55**, 111 (2012).

[224] S. Bubeck, "Convex Optimization: Algorithms and Complexity", Foundations and Trends® in Machine Learning **8**, 231 (2015).

[225] S. P. Boyd and L. Vandenberghe, *Convex optimization* (Cambridge university press, 2004).

[226] Y. Bengio, N. Roux, P. Vincent, O. Delalleau, and P. Marcotte, "Convex neural networks", in Advances in neural information processing systems, Vol. 18, edited by Y. Weiss, B. Schölkopf, and J. Platt (2005).

[227] G. B. Dantzig, "Origins of the simplex method", in *A history of scientific computing* (ACM, New York, NY, USA, 1990), pp. 141–151.

[228] E. H. Ebert, "A Comparison of the Original and Revised Stanford-Binet Scales", The Journal of Psychology **11**, 47 (1941).

[229] D. A. Spielman and S.-H. Teng, "Smoothed analysis of algorithms", Journal of the ACM **51**, 385 (2004).

[230] L. Vandenberghe and S. Boyd, "Semidefinite Programming", SIAM Review **38**, 49 (1996).

[231] N. Karmarkar, "A new polynomial-time algorithm for linear programming", Combinatorica **4**, 373 (1984).

[232] F. A. Potra and S. J. Wright, "Interior-point methods", Journal of Computational and Applied Mathematics **124**, 281 (2000).

[233] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd, "Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding", Journal of Optimization Theory and Applications **169**, 1042 (2016).

[234] L. Menniche and D. Benterki, "A logarithmic barrier approach for linear programming", Journal of Computational and Applied Mathematics **312**, 267 (2017).

[235] C. Helmberg, F. Rendl, R. J. Vanderbei, and H. Wolkowicz, "An Interior-Point Method for Semidefinite Programming", SIAM Journal on Optimization **6**, 342 (1996).

[236] J. A. Smolin, J. M. Gambetta, and G. Smith, "Efficient Method for Computing the Maximum-Likelihood Quantum State from Measurements with Additive Gaussian Noise", Physical Review Letters **108**, 070502 (2012).

[237] E. J. Candès, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements", Communications on Pure and Applied Mathematics **59**, 1207 (2006).

[238] G. Kutyniok, "Theory and applications of compressed sensing", GAMM-Mitteilungen **36**, 79 (2013).

[239] C. A. Riofrío, D. Gross, S. T. Flammia, T. Monz, D. Nigg, R. Blatt, and J. Eisert, "Experimental quantum compressed sensing for a seven-qubit system", Nature Communications **8**, 15305 (2017).

[240] A. I. Lvovsky, "Iterative maximum-likelihood reconstruction in quantum homodyne tomography", Journal of Optics B: Quantum and Semiclassical Optics **6**, S556 (2004).

[241] J. Shang, Z. Zhang, and H. K. Ng, "Superfast maximum-likelihood reconstruction for quantum tomography", Physical Review A **95**, 062336 (2017).

[242] J. Řeháček, Z. Hradil, E. Knill, and A. I. Lvovsky, "Diluted maximum-likelihood algorithm for quantum tomography", Physical Review A **75**, 042108 (2007).

[243] B. Qi, Z. Hou, L. Li, D. Dong, G. Xiang, and G. Guo, "Quantum state tomography via linear regression estimation", Scientific reports **3**, 1 (2013).

[244] R. Blume-Kohout, "Optimal, reliable estimation of quantum states", New Journal of Physics **12**, 043034 (2010).

[245] C. Ferrie and R. Blume-Kohout, "Maximum likelihood quantum state tomography is inadmissible", arXiv:1808.01072 (2018).

[246] M. Cramer, M. B. Plenio, S. T. Flammia, R. Somma, D. Gross, S. D. Bartlett, O. Landon-Cardinal, D. Poulin, and Y.-K. Liu, "Efficient quantum state tomography", Nature Communications **1**, 149 (2010).

[247] F. Pan and P. Zhang, "Simulating the Sycamore quantum supremacy circuits", arXiv:2103.03074 (2021).

[248] G. C. Knee, E. Bolduc, J. Leach, and E. M. Gauger, "Quantum process tomography via completely positive and trace-preserving projection", Physical Review A **98**, 62336 (2018).

[249] T. Surawy-Stepney, J. Kahn, R. Kueng, and M. Guta, "Projected Least-Squares Quantum Process Tomography", Quantum **6**, 844 (2022).

[250] R. W. Johnson, *An Introduction to the Bootstrap*, Vol. 23, 2 (CRC press, 2001), pp. 49–54.

[251] C. Granade, J. Combes, and D. G. Cory, "Practical Bayesian tomography", New Journal of Physics **18**, 033024 (2016).

[252] H. Laurell, D. Finkelstein-Shapiro, C. Dittel, C. Guo, R. Demjaha, M. Ammitzböll, R. Weissenbilder, L. Neoričić, S. Luo, M. Gisselbrecht, C. L. Arnold, A. Buchleitner, T. Pullerits, A. L'Huillier, and D. Busto, "Continuous-variable quantum state tomography of photoelectrons", Physical Review Research **4**, 033220 (2022).

[253] M. Schuld, I. Sinayskiy, and F. Petruccione, "The quest for a Quantum Neural Network", Quantum Information Processing **13**, 2567 (2014).

[254] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning", Physical Review A **98**, 032309 (2018).

[255] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin, "General parameter-shift rules for quantum gradients", Quantum **6**, 677 (2022).

[256] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes", Nature Communications **9**, 1 (2018).

[257] S. Bai, J. Z. Kolter, and V. Koltun, "Deep Equilibrium Models", Advances in Neural Information Processing Systems **32** (2019).

[258] J. Lorraine, P. Vicol, and D. Duvenaud, "Optimizing Millions of Hyperparameters by Implicit Differentiation", Proceedings of Machine Learning Research **108**, 1540 (2019).

[259] R. A. Vargas-HernÃ¡ndez, R. T. Chen, K. A. Jung, and P. Brumer, "Fully differentiable optimization protocols for non-equilibrium steady states", New Journal of Physics **23**, 123006 (2021).

[260] O. Di Matteo and R. M. Woloshyn, "Quantum computing fidelity susceptibility using automatic differentiation", Physical Review A **106**, 52429 (2022).

[261] D. Gottesman, A. Kitaev, and J. Preskill, "Encoding a qubit in an oscillator", Physical Review A **64**, 012310 (2001).