Doctorate of Philosophy

Schrödinger's Catwalk

Brian Flynn

University of Bristol

January, 2021

CONTENTS

I CONTEXTUAL REVIEW

1 QUANTUM THEORY 2
- 1.1 Quantum Mechanics . . . 2
  - 1.1.1 Hamiltonians . . . 5
- 1.2 Quantum Information . . . 5
  - 1.2.1 Expectation value . . . 10
- 1.3 Quantum Simulation and Computation . . . 10

2 MACHINE LEARNING 13
- 2.1 Classical machine learning . . . 13
  - 2.1.1 Supervised machine learning . . . 13
  - 2.1.2 Performance metrics . . . 15
  - 2.1.3 Unsupervised machine learning . . . 17
  - 2.1.4 Reinforcement learning . . . 18
- 2.2 Quantum machine learning . . . 18
- 2.3 genetic algorithms . . . 19
  - 2.3.1 Example: knapsack problem . . . 20
  - 2.3.2 Selection mechanism . . . 23
  - 2.3.3 Reproduction . . . 25
  - 2.3.4 Candidate evaluation . . . 26

Appendix

A FUNDAMENTALS 28
- A.1 Linear algebra . . . 28
- A.2 Postulates of quantum mechanics . . . 29
- A.3 States . . . 30
  - A.3.1 Mulitpartite systems . . . 31
  - A.3.2 Registers . . . 32
- A.4 Entanglement . . . 33
- A.5 Unitary Transformations . . . 33
- A.6 Dirac Notation . . . 34

LIST OF TABLES

Table 2.1 Classification metrics 15
Table 2.2 F_1-score examples. 17
Table 2.3 Candidate solutions to knapsack problem 23
Table 2.4 Genetic algorithm parent selection database 26
Table A.1 Linear algebra defintions 28

LIST OF FIGURES

Figure 1.1 Bloch sphere representation of bases 7
Figure 1.2 Rotations on Bloch sphere 9
Figure 1.3 Expectation values 9
Figure 2.1 Knapsack problem 20
Figure 2.2 Roulette wheels for selection 24
Figure 2.3 Crossover and mutation of chromosomes. 25

LISTINGS

ACRONYMS

ML	machine learning. 11
QM	quantum mechanics. 2–4, 6, 7, 13, 23
QMLA	Quantum Model Learning Agent. vi

GLOSSARY

expectation value	Average outcome expected by measuring an observable of a quantum system many times, Section 1.2.1.. i, 2, 9
likelihood	Value that represents how likely a hypothesis is.. 9

Part I

CONTEXTUAL REVIEW

1 QUANTUM THEORY

Quantum mechanics (QM) – the study of nature's fundamental processes, manifest in its smallest particles – has been at the forefront of physics since the early 20th century [1]. Advances in theoretical understanding of quantum mechanical systems in the first half of the century [2, 3, 4, 5, 6] which came to underpin all modern information and communications technologies [7, 8]. The 21st century, on the other hand, is poised to see the development of technologies which *deliberately* exploit the most intricate quantum processes in order to yield some non-classical advantage. This thesis focuses on the application of machine learning to the characterisation of quantum mechanical systems through use of quantum simulators, so it is pertinent first to introduce the vocabulary of QM.

QM is central to the topics described in this thesis, however it is impossible to succinctly capture the entire discipline; in this chapter we will only introduce concepts utilised throughout. For completeness, we elucidate some fundamental topics of linear algebra and quantum theory in Appendix A, but consider them too cumbersome to include in the main text. For a more complete and general introduction to QM, the reader is referred to [9, 10]. Likewise, in this chapter we quickly summarise the key aspects of quantum computation, but for further details, we recommend unfamiliar readers to consult [11], while a more complete discussion is presented in [12].

1.1 QUANTUM MECHANICS

At any time, a quantum system, Q, can be described by its *wavefunction*, $\Psi(t)$, which contains all information about Q. In analogy with Newton's second law of motion, which allows for the determination of a particle's position at any time, $\vec{r}(t)$, given its conditions such as mass and acceleration as well as its initial position, $\vec{r}(t_0)$, quantum *equations of motion* can describe the evolution of Q through its wavefunction [13]. One proposal[1] for the equation of motion to describe the evolution of the wavefunction under known conditions, i.e. determining $\Psi(t)$ from $\Psi(t_0)\ \forall t > t_0$, is *Schrödinger 's equation* [9, 14, 15].

Although the Schrödinger equation is a *postulate* of QM (see Appendix A.2), let us introduce it in reverse order to elucidate its meaning, following [10]. We have yet to describe the structure of the wavefunction, which we will do in Section 1.2, but here we will represent wavefunctions using *Dirac notation* (Appendix A.6), and can think of them generically as vectors, i.e. $\Psi(t) \to |\psi(t)\rangle$. Suppose we have two such wavefunctions, $|\phi(t)\rangle, |\psi(t)\rangle$ which are functions of time $t > t_0$.

0 Colloquially referred to as *Quantum 1.0*.

1 The most noteworthy alternative formalism, due to Heisenberg [5], was shown equivalent to the Schrödinger picture described here.

We start with the assumption that *similarity* is conserved between two wavefunctions, if they undergo the same transformation (Susskind's *minus first* law of classical mehcanics [10])

$$\langle \phi(t)|\psi(t)\rangle = \langle \phi(t_0)|\psi(t_0)\rangle \tag{1.1}$$

Then, assuming some equations of motion capture the dynamics of Q, there exists some evolution operator, $\hat{U}(t)$, which deterministically maps $|\psi(t_o)\rangle$ to $|\psi(t)\rangle$.

$$|\psi(t)\rangle = \hat{U}(t)\,|\psi(t_0)\rangle\,, \tag{1.2}$$

where we have not yet imposed any restrictions on $\hat{U}$. Combining Eqs. (1.1) to (1.2),

$$\begin{aligned} \langle \phi(t)|\psi(t)\rangle &= \langle \phi(t_0)|\hat{U}^\dagger\hat{U}|\psi(t_o)\rangle \\ \Rightarrow \langle \phi(t_0)|\hat{U}^\dagger(t)\hat{U}(t)|\psi(t_o)\rangle &= \langle \phi(t_0)|\psi(t_0)\rangle \\ \Rightarrow \hat{U}^\dagger(t)\hat{U}(t) &= \hat{\mathbb{1}} \quad \forall t, \end{aligned} \tag{1.3}$$

where the result $\hat{U}^\dagger(t)\hat{U}(t) = \hat{\mathbb{1}}$ is the condition for *unitarity* (Appendix A.5), so we can claim the quantum wavefunction evolves unitarily.

By construction, we require that after zero time, i.e. $t = t_0$, the wavefunction has not changed:

$$\begin{aligned} |\psi(t = t_0)\rangle = \hat{U}(t = t_0)\,|\psi(t_0)\rangle = |\psi(t_0)\rangle \\ \Rightarrow \hat{U}(t = t_0) = \hat{\mathbb{1}}. \end{aligned} \tag{1.4}$$

Without loss of generality we can set $t_0 = 0$, giving $\hat{U}(0) = \hat{\mathbb{1}}$. Then, let us consider an infintesimally small time increment $t_0 + \epsilon$: again, take $t_0 = 0$ so $t = \epsilon$, where $\epsilon \gg \epsilon^2$.

We can say

$$\hat{U}(\epsilon) = \hat{\mathbb{1}} + \mathcal{O}(\epsilon), \tag{1.5}$$

which merely suggests that the time evolution operator at very small time is very close to the identity, with some small displacement proportional to the time, which must be an operator to act on the wavefunction (vector). We suppose the form of the offset, so we can write

$$\hat{U}(\epsilon) = \hat{\mathbb{1}} - \epsilon\left(\frac{i}{\hbar}\hat{H}_0\right), \tag{1.6}$$

where the inclusion of the phase $-i$ is arbitrary, and we have named as $\hat{H}_0/\hbar$ the operator by which the time evolution differs from the identity. In other words, the operator $\hat{H}_0$ is generically the generator of the evolution/dynamics of Q: any difference between $|\psi(t_0)\rangle$ and $|\psi(t)\rangle$ arises

solely due to $\hat{H}_0$. So far there is no restriction on $\hat{H}_0$, except that it must be of the same dimension as the Hilbert space in question. Recalling the unitarity condition, however:

$$\begin{aligned} \hat{U}^\dagger(\epsilon)\hat{U}(\epsilon) &= \hat{\mathbb{1}} \\ \Rightarrow \left(\hat{\mathbb{1}} + \frac{i}{\hbar}\epsilon\hat{H}_0^\dagger\right)\left(\hat{\mathbb{1}} - \frac{i}{\hbar}\epsilon\hat{H}_0,\right) &= \hat{\mathbb{1}} \\ \Rightarrow \hat{\mathbb{1}} + \frac{i}{\hbar}\epsilon(\hat{H}_0^\dagger - \hat{H}_0) + \mathcal{O}(\epsilon^2) &= \hat{\mathbb{1}} \\ \Rightarrow (\hat{H}_0^\dagger - \hat{H}_0) &= 0 \\ \Rightarrow \hat{H}_0^\dagger &= \hat{H}_0. \end{aligned} \tag{1.7}$$

Eq. (1.7) results in the condition for *Hermiticity*, meaning that $\hat{H}_0$ is an observable of Q. In fact, this is the *Hamiltonian* of the system, described in the next section.

We can also use the infintesimal evolution to see

$$\begin{aligned} |\psi(t)\rangle &= \hat{U}(t)\,|\psi(t_0)\rangle \\ \Rightarrow |\psi(\epsilon)\rangle &= \hat{U}(\epsilon)\,|\psi(t_0)\rangle \\ \Rightarrow |\psi(\epsilon)\rangle &= \left(\hat{\mathbb{1}} - \epsilon\frac{i}{\hbar}\hat{H}_0\right)|\psi(t_0)\rangle \\ \Rightarrow |\psi(\epsilon)\rangle &= |\psi(t_0)\rangle - \epsilon\frac{i}{\hbar}\hat{H}_0\,|\psi(t_0)\rangle \\ \Rightarrow \frac{|\psi(\epsilon)\rangle - |\psi(t_0)\rangle}{\epsilon} &= -\frac{i}{\hbar}\hat{H}_0\,|\psi(t_0)\rangle \end{aligned} \tag{1.8}$$

Taking the limit as $\epsilon \to 0$, the left hand side of the final line of Eq. (1.8) is the definition of the derivative of the wavefunction, $\frac{d|\psi(t)\rangle}{dt}$. Taken together, we have

$$\frac{d}{dt}\,|\psi(t)\rangle = \frac{-i}{\hbar}\hat{H}_0\,|\psi(t_0)\rangle\,, \tag{1.9}$$

where $|\psi(t)\rangle$ is the wavefunction at time t, $|\psi(t_0)\rangle$ is the wavefunction at t_0, such that $t > t_0$, $\hbar = 1.054 \times 10^{-34}$ is the reduced Planck constant and $\hat{H}_0$ is the *Hamiltonian* of Q. For brevity we generally refer to $t_0 = 0$, and absorb $\hbar$ into $\hat{H}_0$, which will later manifest in the Hamiltonian scalar parameters. Eq. (1.9) is the most general form of *Schrödinger equation*, otherwise known as the *time-dependent* Schrödinger equation; we include it as Postulate 6 when describing the fundamentals of QM (Appendix A.2), since it can be seen as an irreducible equation of motion which is essential to the description of quantum systems.

As mentioned, we presented this argument in a nonstandard order: we started with Eq. (1.2), which we can now consider the *solution* to the Schrödinger , specifically

$$\begin{aligned} |\psi(t)\rangle &= \hat{U}(t)\,|\psi(0)\rangle \\ \hat{U}(t) &= e^{-i\hat{H}_0 t} \end{aligned} \tag{1.10}$$

which describes the unitary evolution of the state according to the unitary evolution operator, $\hat{U}(t)$.

1.1.1 *Hamiltonians*

In the previous section we introduced the Hamiltonian[2] of Q as the generator of its time evolution dynamics; Hamiltonians are of primary importance in this thesis, so it is worth pausing to consider their physical meaning. We saw in Eq. (1.7) that $\hat{H}_0$ is Hermitian, meaning that the operator is physically observable according to Postulates 2 to 3 of quantum mechanics (Appendix A.2). The Hamiltonian operator captures the energy of Q: the eigenvalues of the observable $\hat{H}_0$ are the permitted energy levels of the system.

The quantum Hamiltonian, $\hat{H}_0$, is analogous to the classical Hamiltonian, insofar as it captures all the interactions of a given system which contribute to its time evolution. Knowing the classical Hamiltonian and the initial conditions – position and momentum – Hamilton's equations of motion allow for the calcaultion of those quantities for the particle in question an infintesimal time later [16]. Likewise, knowledge of the initial wavefunction, $|\psi(t_0)\rangle$, and the system's quantum Hamiltonian, $\hat{H}_0$, the quantum equations of motion – Schrödinger 's equation, Eq. (1.9) – permits the calculation of the wavefunction at later times. As such the Hamiltonian must consist of all processes which influence the evolution of Q; we will later break the Hamiltonian into independent *terms* which each correspond to unique physical interactions Q is subject to, **??**. We can think that each process/interaction Q undergoes contributes to its total energy, giving intuition as to why its eigenvalues are the energy levels.

Hamiltonians describe *closed* quantum systems, i.e. where *all* processes and interactions which influence Q are accounted for. Realistic quantum systems are influenced by a myriad of proximal systems, and it is therefore infeasible to analytically account for them all. Instead, *open* quantum systems' dynamics are described by Lindbladian operators, which encompass the Hamiltonian form. The Lindblad master equation is a generalisation of the Schrödinger equation, providing the equation of motion for open quantum systems [17, 18]. In this thesis we only consider closed models for quantum systems; for meaningful impact of the techniques presented here, it will be necessary to expand them to account for the open system dynamics of realistic experiments. We do, however, show initial progress towards this endeavour by modelling a physical system through a closed Hamiltonian, **??**.

1.2 QUANTUM INFORMATION

We have not yet described the structure of the wavefunction, instead performing the previous analysis with respect to some arbitrary objects. The wavefunction for a physical system, Q,

2 Aside: the author shares a hometown with the mathematician for whom it is named, William Rowan Hamilton. It is hoped that, after another 150 years, the next physicist from Trim, Co. Meath, Ireland might profitably use knowledge of Hamiltonians on a quantum computer (QC).

is also known as its *state*, a complete mathematical description of the system [19]. States are vectors[3], $|\psi\rangle \in \mathbb{C}$; the valid state space for Q is its *Hilbert space*, $\mathcal{H}$, which is a generalisation of Euclidean vector space, i.e. $|\psi\rangle \in \mathcal{H}$. The Hilbert space defines the overlap between any two vectors as the *inner product*, $\langle\psi|\phi\rangle$ (Appendix A.1). In general[4], a state can be seen as a *superposition* across its eigenstates, $\{|v_i\rangle\}$.

$$|\psi\rangle = \sum_i \alpha_i |v_i\rangle \tag{1.11a}$$

$$\text{subject to} \quad \sum_i |\alpha_i|^2 = 1, \quad \alpha_i \in \mathbb{C}. \tag{1.11b}$$

The cornerstone of QM is the effect of *measurement* on quantum systems: in general Q can be seen as occupying a multitude of eigenstates as in Eq. (1.11a); observing the system forces $|\psi\rangle$ into a definite occupation of a single eigenstate, where the *probability* that it is measured in each eigenstate $|v_i\rangle$ is given by $|\alpha_i|^2$, according to Born's rule [3]. α_i are hence named *probability amplitudes* since they inform the probability of measuring the corresponding eigenstate.

For an ideal[5] single particle, when the state, Eq. (1.11a), has two available eigenstates, e.g. the horizontal (H) and vertical (V) polarisation of a single photon, we can designate Q as a two-level computational platform, called a *qubit*, analogous to the workhorse of classical computation, the bit. A qubit's state vector can then be written as a sum over the two available eigenstates, where we assign vectors to the eigenstates as $|H\rangle = |v_1\rangle, |V\rangle = |v_2\rangle$.

$$|\psi\rangle = \alpha_1 |v_1\rangle + \alpha_2 |v_c\rangle, \tag{1.12}$$

where $\alpha_i \in \mathbb{C}$ and $|\alpha_1|^2 + |\alpha_2|^2 = 1$.

In general, a qubit requires two orthogonal state vectors to define a *basis*; we list a number of the usual special cases:

$$X\text{-basis} = \begin{cases} |+\rangle = \dfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |-\rangle = \dfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{cases} \tag{1.13a}$$

$$Y\text{-basis} = \begin{cases} |i\rangle = \dfrac{1}{\sqrt{2}}(|0\rangle + i|1\rangle) \\ |-i\rangle = \dfrac{1}{\sqrt{2}}(|0\rangle - i|1\rangle) \end{cases} \tag{1.13b}$$

3 We immediately use Dirac notation to represent the state; it is defined in Appendix A.6.
4 We expand on this brief description in Appendix A.3.
5 Here we restrict to the space of ideal, *logical* qubits. In reality, physical qubits are beset by errors, demanding error correction routines such that multiple particles are needed attain a single logical qubit.

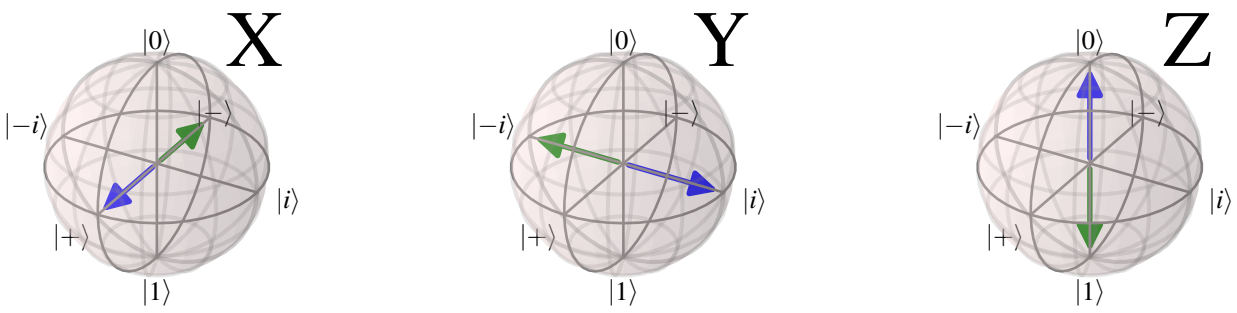

Figure 1.1: Bloch sphere representation of bases, where each pair of basis states are shown by blue and green vectors. The X-basis has basis vectors $\{|+\rangle, |-\rangle\}$; Y-basis has $\{|i\rangle, |-i\rangle\}$ and Z-basis has $\{|0\rangle, |1\rangle\}$.

$$Z\text{-basis} = \begin{cases} |0\rangle \\ |1\rangle \end{cases} \tag{1.13c}$$

A visual tool for representing qubits is the *Bloch sphere*, which presents orthogonal basis states as parallel unit vectors of opposite direction: we show each of the bases of Eq. (1.13) in Fig. 1.1.

We can make two remarks about basis states for a single qubit:

- Basis states from one basis can be seen as superpositions with respect to alternative bases
  - e.g. in the X-basis, $|+\rangle$ is a basis vector, but in the Z-basis, $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ is a superposition over basis vectors.
- Bases are local rotations of each other
  - rotating the X-basis through an angle $\pi/2$ about the Y-axis results in the Z-axis.

As we alluded to in Section 1.1: by imposing mathematical structure on quantum systems' states, i.e. representing Q as a state vector at any time, then operations which alter the state of the system must be matrices, which we will call *operators*. In general an n-dimensional vector is rotated by an $n \times n$ matrix; therefore to rotate the one-qubit state, given by a two-dimensional vector, we require a 2×2 operator. One-qubit operators have the effect of rotating the state vector, which we can again visualise on the Bloch sphere. By thinking of qubits generically with respect to any basis, we can encode information in the qubit's amplitudes, by performing operations (or gates) upon the qubit, we change the information, i.e. we can design information processing techniques leveraging the infrastructure – states, operators and measurement – of QM.

We introduce a set of speical one-qubit operators, the *Pauli matrices*,

$$\hat{\sigma}_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{1.14a}$$

$$\hat{\sigma}_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{1.14b}$$

$$\hat{\sigma}_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{1.14c}$$

The Pauli matrices are used to define rotation operators about their respective axes, and hence are very useful: we can break *any* rotation of a qubit into rotations of various angles, θ, about the three axes of the Bloch sphere. Any single qubit operation can therefore be expressed as a product of the *rotation operators*, $\hat{R}_x, \hat{R}_y, \hat{R}_z$, exemplified in Fig. 1.2 and defined for $w \in \{x, y, z\}$ as

$$\hat{R}_w(\theta) = e^{-i\frac{\theta}{2}\hat{\sigma}_w} = \cos(\theta/2)\,\hat{\mathbb{1}} - i\sin(\theta/2)\,\hat{\sigma}_w. \tag{1.15}$$

The Pauli matrices are Hermitian, meaning they are observable. We can see that the eigenstates of $\hat{\sigma}_z$ are the Z-basis states: $\hat{\sigma}_z|0\rangle = |0\rangle\,; \hat{\sigma}_z|1\rangle = -|1\rangle$. Recalling the earlier claim that the two-level quantum system (e.g. H and V polarisation of a photon) can be mapped to eigenvectors[6] of an obserable operator to form a qubit, we term the Z-basis the *computational* basis. By defining the computational basis, we ground abstract computational reasoning in the physical realisation: anywhere throughout this thesis where the basis states $|0\rangle, |1\rangle$ are referenced, we mean the eigenstates of the physical axis which is defined as the Z-axis for the system in question. In the computational basis, then, a qubit can be specified as

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle. \tag{1.16}$$

The concepts of qubits representing quantum systems, as well as operators altering their states and measurement collapsing those states, extend straightforwardly to multipartite systems, by merging Hilbert spaces through tensor products, as we show in Appendix A.3.1. The cases where this is a simple mathematical exercise represent *pure* states; of course this leads to the more intricate topic of *mixed* states and entanglement, briefly described in **??**. In this thesis, however, we are concerned only with pure states, i.e. separable qubits. While single qubit states are spanned by the Pauli opeartors, multi-qubit states are spanned by the Pauli group, $\mathbb{G}$: n-qubit states are spanned by $\mathbb{G}_n = \left(\mathbb{C}^2\right)^{\otimes n}$.

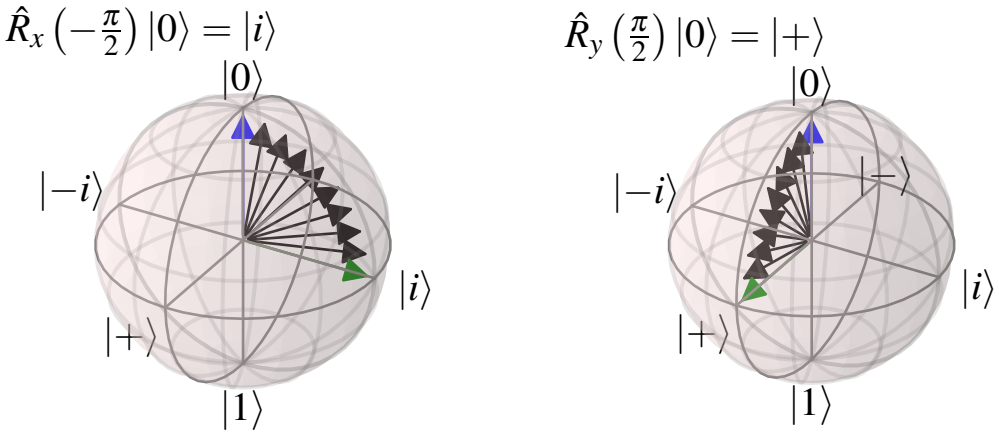

Figure 1.2: Rotations on Bloch sphere. The initial and final state are shown in blue and green respectively, while intermediate states are shown in black. **Left**, The Z-basis unit vector, $|0\rangle$, is rotated about the X-axis, resulting in the unit vector along the Y-axis. **Right**, The Z-basis unit vector, $|0\rangle$, is rotated about the Y-axis, resulting in the unit vector along the X-axis.

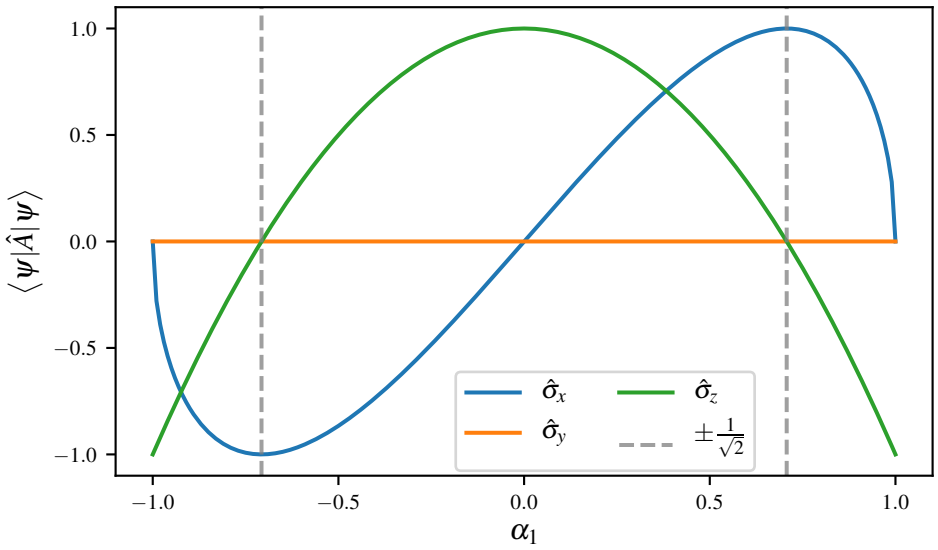

Figure 1.3: Expectation values of the observable $\hat{A}$ – here the Pauli matrices – for $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$. Coefficients are real, varying $\alpha_1 \in (-1, 1)$, such that $\alpha_0 = \sqrt{1-\alpha_1^2}$.

1.2.1 *Expectation value*

Upon measurement, the state vector of Q has amplitude associated with only one eigenvector. On average, however, the eigenvector to which it would collapse encodes statistical insight on the state prior to measurement. In other words, if we prepared $|\psi\rangle$ and measured it – via some observable, $\hat{A}$ – and repeat the procedure N times, then as $N \to \infty$, the average outcome is the *expecatation value* for the system is given by

$$\langle \hat{A} \rangle = \langle \psi | \hat{A} | \psi \rangle = \sum_i \alpha_i \langle v_i | \hat{A} | v_i \rangle \,, \tag{1.17}$$

where $\langle \hat{A} \rangle$ is the expectation value (average) for the observable, $\hat{A}$; $|v_i\rangle$ are the eigenvectors of $\hat{A}$, and $\alpha_i \in \mathbb{C}$ are the proability amplitudes associated with each $|v_i\rangle$ when the state $|psi\rangle$ is represented as in Eq. (1.11a). We show some examples of expectation values for the observable Pauli matrices in Fig. 1.3.

An underlying theme of this thesis is to flip the usual logic: instead of using knowledge of the system to derive the expectation value, per Eq. (1.17), we will *estimate* expectation values, either through experiment or simulation, and use them to infer the structure of the observable. This trick enables machine learning routines to reverse engineer the processes Q is subject to, as we will describe in **??**. There is a subtlety to be aware of, however: in later chapters, we will make reference to the expectation value and *likelihood* almost interchangeably – these are distinct concepts that *sometimes* overlap, but not in all cases, as we explain in **??**.

1.3 QUANTUM SIMULATION AND COMPUTATION

Relying on the premise and language of quantum information processing – states, qubits, operators, measurements and expectation values – the growing field of *quantum technology* aims to exploit the non-classical statistics yielded by quantum systems in order to retrieve outcomes beyond the capability of their classical counterparts [20]. Applications range from enhanced sensing and metrology [21, 22], to highly-secure communication and cryptography protocols [23, 24, 25]. The initial motivation for the development of quantum technologies, however, was the observation that simulating nature at a quantum level would require exponential resources and is therefore only feasible given controllable quantum systems, which can accurately emulate the true dynamics [26, 27, 28, 29].

The notion of controlling quantum systems to mimic the dynamics of real quantum systems is tantamount to *quantum simulation* [30, 31]. In particular, simulating quantum systems is believed to be of interest for quantum chemistry [32, 33, 34], for example leading to advances in the simulation of molecular dynamics [35, 36]. More generally, however, this led to research into a wider domain of calculations called *quantum computation* which considers the information

6 The terms *eigenstate* and *eigenvector* are interchangeable, although it may be helpful to think of eigenstate as the physical manifestation (horizontal photon), and think of eigenvector as the logical manifestions ($|0\rangle$).

processing capability of controllable quantum systems beyond merely simulating quantum systems. Then, *universal quantum computers* (or, *quantum Turing machines*), assume access to *logical* qubits and operations (or *gates*) for the implementation of quantum circuits [37]. This ignited interest in *quantum algorithms*, which aim to provide some provable advantage [38, 39, 40, 41, 42]. Indeed, it was found that the space of problems addressible by such devices goes beyond the classical counterpart, suggesting there exists a class of quatum algorithms which can offer significant advantage over any feasible algorithm on classical hardware [43].

Of course, while the advances in algorithmic quantum computation promise huge impact, they are tempered by contemporary experimental constraints, which must deal with the reality that construction and control of quantum devices is a significant challenge. In constructing QCs and dealing with their output, we must account for physical effects which lead to errors, requiring expensive error mitigation schemes to be reliable [44, 45]. Furthermore, there are a number of criteria a QC must meet before it can be deemed reliable [46].

Any two-level quantum system can be used as a qubit; a range of platforms have emerged in attempts to fulfil the potential of quantum computation [47], among the most popular[7]:

- Photonic qubits (linear optical QCs) [48]
  - existing infrastructure for commercial production of photonics-based technologies suggests the relatively straightforward fabrication of integrated photonic devices at the scale of millions of degrees of freedom [49];
  - photons do not decohere so are useful for encoding information [50];
  - photons do not interfere [13, 51], so information processing must be mediated by non-trivial measurement schemes [52];
  - they are liable to a unique error mechanism – photon loss – necessitating novel quantum error correcting codes [53];
  - on-demand single photon generation has not yet been demonstrated, although there is significant progress in the area of photon generation [54].
- Superconducting qubits [55, 56]
  - relatively straightforward to control and couple with each other, enabling high-fidelity two-qubit gates, e.g. 99.7% reached in [57];
  - difficult to engineer substantial coherence times, although there has been significant recent progress [58, 59], e.g. $T_1 \sim \mathcal{O}(\text{ms})$ in [60];
  - require cryogenic temperatures [55];
  - arbitrary qubit connectivity at scale is yet to be demonstrated [56];
  - methods for the fabrication of medium-to-large scale devices required for fault tolerant quantum computation are not yet known [61].
- Ion traps [62, 63]

7 An incomplete list of quantum architectures together with their primary advantages and limitations.

  - high two qubit gate fidelities, e.g. 99.9% in [64];
  - very high coherence times, e.g. $\mathcal{O}(10\text{ min})$ in [65];
  - straightforward state preparation and readout, e.g. 99.99% readout fidelity in [66];
  - long gate-times, e.g. $\mathcal{O}(\mu\text{s})$ in [67];
  - uncertain scalability [68].

The ever-increasing space of contenders has led to a growing eco-system for quantum software [69], promising a wide range of applications in the era of noisy intermediate scale quantum devices [70]. Following numerous proposoals [71], recent efforts have married state-of-the-art hardware with bespoke algorithms in order to achieve quantum advantage [72, 73]. Evidently there is vast effort in bringing quantum computational resources to reality; in this thesis, however, we are not concerned with the architecture underlying our presumed quantum simulator – we perform simulations only on classical hardware. In principle, however, any quantum simulator – universal or otherwise – capable of implementing the time evolution operator, Eq. (1.10), can be called upon as a co-processor by the algorithms presented.

Our restriction to classical resources leads to a few remarks:

- given access to a fault-tolerant QC/simulator, the algorithms described would enjoy an automatic exponential speedup:
  - the classical bottleneck is the calculation of the time evolution dynamics, Eq. (1.10), according to the matrix exponential, of dimension 2^n, where n is the dimension of the system;
  - it is believed that the same calculation can be performed in polynomial time on a QC [30].
- the results achieved in this thesis are limited by the capability of classical computers in simulating quantum systems
  - we study only up to 8-qubit systems, whereas it would be of interest to extend these methods to higher dimensions, which is expected to be feasible when reliable quantum simulators/computers are available.

The remit of this thesis – given these limitations – is therefore to robustly test the presented algorithms, and provide a benchmark achieved through classical facilities, against which the same algorithm can be run in conjunction with quantum hardware.

2

MACHINE LEARNING

Machine learning (ML) is the application of statistics, algorithms and computing power to discover meaning and/or devise actions from data. ML has become an umbrella term, encompassing the family of algorithms which aim to leverage computers to learn without being explicitly programmed, as opposed to the more general artificial intelligence (AI), which seeks to make computers behave intelligently, admitting explicit programs to achieve tasks [74]. Its history is therefore imprecise since a number of early, apparently unrelated algorithms were proposed independently, which now constitute ML routines [75, 76]. Nevertheless, the field of ML has been advancing rapidly since the second half of the 20$^{\text{th}}$ century [77], especially recently due to the availability of advanced hardware such as graphics processing units (GPUs), facilitating significant progress through an ever-increasing aresnal of powerful open source software [78, 79, 80].

Throughout this thesis, we endeavour to combine known methods from the ML literature with capabilities of QCs[1]. Typical ML algorithms, which rely on central processing units (CPUs) or GPUs, are deemed *classical* machine learning, in contrast with quantum machine learning (QML), where QCs are central to processing the data. Similarly to the remit of Chapter 1, here we do not provide an exhaustive account of ML algorithms: rather, we describe only the algorithms which are used in later chapters, referring readers to standard texts for a wider discussion [77, 81].

2.1 CLASSICAL MACHINE LEARNING

Of course the first step in any ML application is to consider the types of available data, with respect to the ensemble of known algorithms. Classical ML is usually described in three categories, broadly based on the format of data on which the insight can be built; we will briefly describe each to provide context to discussions throughout this thesis. Later in this thesis we will use the word *model* for descriptions of quantum systems, but here *model* refers to the mapping between inputs and outputs which the ML algorithm devises.

2.1.1 *Supervised machine learning*

Models are trained using *labelled* data, i.e. each training sample has a known label y_i; or, a set of *feature vectors* $\{\vec{x}_i\}$ are associated with the set of corresponding *classes* $\{y_i\}$ [82]. The output is a predictive tool which aims to reconstruct the classes of unseen feature vectors: in general, we can view the role of ML in this setting as distilling the function f such that

$$f : \vec{x}_i \longmapsto y_i \quad \forall(\vec{x}_i, y_i). \tag{2.1}$$

1 Or simulated QCs, in this thesis.

There are a number of families of algorithms even within the broad category of supervised ML, we define them as

- *Classification*: aim to produce models which can assign unseen instances to the most appropriate label, from a fixed set of available labels [83];
  - e.g. labels indicate animals' species, and the feature vector for each sample (data point) encodes the animals' height, weight, number of legs, etc.
- *Regression*: models capture the formulaic relationship – which can be either linear or polynamial – between numerical features a the target scalar value, by determining coefficients for each feature.
  - e.g. y is the salary of employees in a company, and the feature vector consists of the individual employees' age, seniority, experience in years, etc.
- *Neural networks*[2]: *universal function approximators*. By invoking a set of linear and non-linear transformations on input data, the *network* is a function of some paramterisation w; neural networks aim to find the optimal network, w', such that Eq. (2.1) is satisfied, $f(w') : \vec{x}_i \longmapsto y_i$.
  - Usually used for classification.
  - e.g. input *neurons*[3] encode the pixel values of images. The neural network can be used to classify the objects it detects within the image.
- *Support vector machine (SVM)*: distinguishes similar data points by projecitng data into higher dimensional space, and therein finding the hypersurface which separates classes [85].
  - Usually used for classification tasks.
  - For data $\mathcal{D} \in \mathbb{R}^n$, a hypersurface in $\mathbb{R}^{n-1}$, can be drawn arbitrarilily through the space (e.g. a 2D plane in a 3D space).
  - Unseen data can then classified by which partition they are reside in when projected into the n-dimensional space.
  - The task of the SVMs is to orient the hypersurface in such a way as to separate distinct classes.

Supervised ML algorithms rely on the existence of a body of labelled samples – the dataset $\mathcal{D}$ – upon which the model can be trained. Training is typically performed on a subset of the data, $\mathcal{D}_t$, usually 80% of samples chosen at random. The remaining (20%) of samples, $\mathcal{D}_v$, are retained for the evaluation of the resultant model: $d \in \mathcal{D}_v$ are not trained upon, so do not contribute to the structure of f as returned by the algorithm. The model therefore can not *overfit*, i.e. simply

2 Including deep learning networks [84].

3 The term *neuron*, also known as *node* or *unit*, derives from the motivation for this class of algorithm: the cells in the brain used for processing information.

recognise particular samples and label them correctly, without any meaningful inference. The evalutation thus captures how the model can be expected to perform on future, unlabelled data.

2.1.2 *Performance metrics*

An important step is the fair assessment of algorithms, then, which is achieved through a number of *performance metrics*. In each supervised ML, the machine attempts to learn the structure of f that optimises some internal objective function (OF), e.g. minimising the average distance between predicted and target labels for regression, $\sum_{d \in \mathcal{D}_t} y_d - y'_d$. To assess the resultant model, we introduce a number of *performance metrics*, which aim to measure serveral perspectives of the model's efficacy, by considering the model's predictions with respect to $\mathcal{D}_v$.

By definition of the data format, it is relatively straightforward to define metrics for supervised routines: the classes assigned to feature vectors, y'_i, can be quantitatively assessed with respect to their true class, y_i. For example, in *binary classification*, the output of the model is either correct or incorrect, allowing us to meaningfully assess its average performance. Likewise, for numerical targets, the difference $\left|y_i - y'_i\right|$ between the output, y'_i, and the target, y_i for each sample cumulatively indicate the strength of the model.

There are a large number of quantities and performance metrics against which to judge models' outputs. In binary classification, we care about whether the model predicts that a given feature is present, and whether it predicts features incorrectly. For example, the model aims to classify whether or not a dog is present in an image. These type of binary predictions have four outcomes:

	y has feature	y' has feature
True positives (TP)	✓	✓
False positives (FP)	×	✓
True negatives (TN)	×	×
False negatives (FN)	✓	×

Table 2.1: Classification metrics. We define classification outcomes based on whether the considered feature was present and/or predicted.

We can use the concepts of Table 2.1 to define a series of *rates* which characterise the model's predictions.

ACCURACY The overall rate at which the algorithm predicts the correct result.

$$\frac{TP + TN}{TP + TN + FN + FP} \tag{2.2a}$$

PRECISION Positive predicitve rate. Of those *predicted to have* the feature of interest, what percentage *actually have* the feature.

$$\frac{TP}{TP + FP} \tag{2.2b}$$

SENSITIVITY True positive rate (also known as *recall*). Of those which *actually have* the feature, what percentage are *predicted to have* the feature.

$$\frac{TP}{TP + FN} \tag{2.2c}$$

SPECIFICITY True negative rate. Of those which *do not actually have* the feature of interest, how many are *predicted not to* have the feature.

$$\frac{TN}{TN + FP} \tag{2.2d}$$

Each metric has clear advantages, but consider also their drawbacks:

- Accuracy can be extremely misleading. for example, in a dataset with only 100 instances of the feature of interest, a model predicting 9,900 true negatives and 1,100 false negatives will give 99% accuracy, despite not having found a single positive sample. This is clearly of no use in identifying the minority of cases which are of actual interest.
- Sensitivity can be inflated by over-fitting to positive cases. That is, by predicting the feature as present in all cases, all true instances of the feature will be found, however all false instances will be labelled as having the feature, so the model has not helped separate the data. The model willl yield a high rate of TP but also a high rate of FP.
- Precision can be high for extremely selective models, i.e. those which are conservative in predicting the presence of the feature. By predicting relatively few positive instances, it can ensure that a high proportion of its predictions are correct. The absolute number of instances identified, however, is relatively low, as a proportion of the total number in the dataset.
- Specificity, similar to sensitivity, can easily mislead by identifying very few instances as having the target feature. Then, it will correctly predict most non-present instances as false, but will not identify the few instance of interest.

Clearly, the performance metric must be chosen with due consideration for the application; e.g. in testing for a medical condition, rather than incorrectly telling a patient they do not have the condition because the model predicted they were feature-negative, it is preferable to incorrectly identify some patients as feature-positive (since they can be retested). In this case, high accuracy is crucial, at the expense of precision. It is clearly appropriate to blend together the usual metrics, in order to derive performance metrics which balance the priorities of the outcomes. In general –

True positives	False negatives	False positives	Precision	Sensitivity	F_1-score
500	500	1000	33	50	$(\frac{2\times33\times50}{33+50}) = 37$
500	500	500	50	50	$(\frac{2\times50\times50}{50+50}) = 50$
1000	0	1000	50	100	$(\frac{2\times50\times100}{50+100}) = 67$
1000	0	0	100	100	$(\frac{2\times100\times100}{100+100}) = 100$

Table 2.2: F_1-score examples.

including in this thesis – the most important aspects of a ML are precision and sensitivity: a model which performs well with respect to *both* of these is sensitive to the feature, but precise in its predictions. A quantity which captures both of these is the F_β-score,

$$f_\beta = \left(1+\beta^2\right)\frac{\text{precision}\times\text{sensitivity}}{(\beta^2\times\text{precision})+\text{sensitivity}}, \tag{2.3}$$

where $\beta \in \mathbb{R}$ is the relative weight of importance between sensitivity and precision. In particular, considering precision and sensitivity as equally important, i.e. $\beta = 1$, we have the F_1-score,

$$f_1 = \frac{2\times\text{precision}\times\text{sensitivity}}{\text{precision}+\text{sensitivity}}. \tag{2.4}$$

For examples of how F_1-score balances these considerations, see Table 2.2.

2.1.3 *Unsupervised machine learning*

Contrary to supervised algorithms, unsupervised methods operate on *unlabelled* data, $\mathcal{D}$. This is often summarised as finding structure within unstructured data. Again we can further compartmentalise methods under this umbrella as follows [86].

- *Clustering*: finding datapoints which are similar to each other, according to some distance metric.
  - e.g. online retailers grouping together customers with similar preferences, in order to tailor advertising campaigns.
- *Dimensionality reduction*: reducing the feature vector of each sample in a dataset to essential components, which may be amalgamations of original features, while retaining structure within the data.
  - this can be used for visualisation to allow for inspection of complex datasets, e.g. plotting users of a social network as nodes on a 2D map, where distinct social groups are kept distant.
- *Association learning*: discover correlations among data.

  - For instance, a supermarket may find that purchasers of certain products are likely also to buy others, providing actionable insight e.g. purchases involving bread also include butter in 50% of cases, so positioning these nearby may increase sales by reminding consumers of their compatibility.
- *Semi-supervised learning*: combine elements of supervised and unsupervised algorithms, to achieve a task beyond the remit of either alone. This often means classifying data where only occur a small subset of the total dataset is labelled.
  - e.g. in facial recognition, a clustering algorithm finds similarities between individual people in photos, and identifies a single person present across a number of photos, and associates those photos together. It combines this with a small set of photos for which people have been tagged, locates the same person and automatically tags them in the wider set of photos.

2.1.4 *Reinforcement learning*

A third category of ML algorithms are reinforcement learning (RL). These are methods where an *agent* interacts with some environment, and refines a *policy* for reacting to different stimuli. As such, the agent can, in principle, deal with a wide array of situations. These methods underly technologies such as self-driving cars, which inspect their surroundings through *sensors*; compute an *action* according to the policy; implement that action through *actuators*. Following an action, the agent senses whether that action was beneficial or detrimental, and receives a *reward* or *punishment* accordingly. Highly rewarded actions are likely to be repeated in future, allowing the agent to learn what response is appropriate to situational parameters, e.g. a self-driving car braking at a red light is rewarded, while braking at a green light is punished.

The concept of a machine's *agency* is an ongoing discussion in the ML community [**?**, **?**], and is important in this thesis also, when we define our model learning protocol as an agent, since it designs models by interacting with the envinroment of the target quantum systems. We will revisit the concept in **??**.

2.2 QUANTUM MACHINE LEARNING

- distinctions
  - q data q hardware $\rightarrow$ pure QML
  - q data classical hardware $\rightarrow$ ml for q physics
  - classical data q hardware $\rightarrow$ q enhanced ml
  - classical data c hardware $\rightarrow$ wrong thesis (Section 2.1)
- examples/applications of QML

  - QNN, q svm,
- Remit of this thesis → ml for q physics
  - i.e. using data from quantum system and/or hardware but in conjunction with classical co-processor, for the study of quantum systems

[87]

2.3 GENETIC ALGORITHMS

In later chapters we will use a class of optimisation techniques known as *evolutionary algorithms* [88, 89]. In particular, *genetic algorithms (GAs)* are central to our primary applications, specifically in **??**. Here we describe genetic algorithms (GAs) in general terms for reference in later chapters.

GAs work by assuming a given problem can be optimised, if not solved, by a single candidate among a fixed, closed space of candidates, called the population, $\mathcal{P}$. A number of candidates are sampled at random from $\mathcal{P}$ into a single *generation*, and evaluated through some objective function (OF), which assesses the fitness of the candidates at solving the problem of interest. Candidates from the generation are then mixed together to produce the next generation's candidates: this *crossover* process aims to combine only relatively strong candidates, such that the average candidates' fitness improve at each successive generation, mimicing the biological mechanism whereby the genetic makeup of offspring is an even mixture of both parents through the philosophy of *survival of the fittest*. The selection of strong candidates as parents for future generations is therefore imperative; in general parents are chosen according to their fitness as determined by the OF. Buidling on this biological motivation, much of the power of GAs comes from the concept of *mutation*: while offspring retain most of the genetic expressions of their parents, some elements are mutated at random. Mutation is crucial in avoiding local optima of the OF landscape by maintaining diversity in the examined subspace of the population.

GAs are not defined either as supervised or unspervised methods; this designation depends on the OF. If candidates are evaluated with respect to labelled data, we can consider that GA supervised, otherwise unsupervised. Pseudocode for a generic GA is given in Algorithm 1, but we can also informally define the procedure as follows. Given access to the population, $\mathcal{P}$,

1. Sample N_m candidates from the population at random
   (a) call this group of candidates the first generation, μ.
2. Evaluate each candidate $\gamma_j \in \mu$
   (a) each γ_j is assigned a fitness, g_j;
   (b) the fitness is computed through an objective function acting on the candidate, i.e. $g_j = g(\gamma_j)$.
3. Map the fitnesses of each candidate, $\{g_j\}$, to selection probabilities for each candidate, $\{s_j\}$
   (a) e.g. by normalising the fitnesses, or by removing some poorly-performing candidates and then normalising.

4. Generate the next generation of candidates
   (a) Reset $\mu = \{\}$;
   (b) Select pairs of parents, $\{\gamma_{p_1}, \gamma_{p_2}\}$, from μ
      i. Each candidate's probability of being chosen is given by their s_j.
   (c) Cross over $\{\gamma_{p_1}, \gamma_{p_2}\}$ to produce children candidates, $\{\gamma_{c_1}, \gamma_{c_2}\}$
      i. mutate $\gamma_{c_1}, \gamma_{c_2}$ according to some random probabilistic process;
      ii. keep γ_{c_i} only if it is not already in μ, to ensure N_m unique candidates are tested at each generation.
   (d) until $|\mu| = N_m$, iterate to step (b).
5. Until the N_g^{th} generation is reached, iterate to step 2.;
6. The strongest candidate on the final generation is deemed the solution to the posed problem.

Candidates are manifested as *chromosomes*, i.e. strings of fixed length, whose entries, called *genes*, each represent some element of the system. In general, genes can have continuous values, although usually, and for all purposes in this thesis, genes are binary, capturing simply whether or not the gene's corresponding feature is present in the chromosome.

2.3.1 *Example: knapsack problem*

One commonly referenced combinatorial optimisation problem is the *knacksack problem*: given a set of objects, where each object has a defined mass and also a defined value, determine the set of objects to pack in a knapsack which can support a limited weight, such that the value of the packed objects is maximised. Say there are n objects, we can write the vector containing the values of those objects as $\vec{v}$, and the vector of their weights as $\vec{w}$. We can then represent configurations of object sets as candidate vectors $\vec{\gamma}_j$, whose genes are binary, and simply indicate whether or not the associated object is included in the set. For example, with $n = 6$,

$$\gamma_j = 100001 \Longrightarrow \vec{\gamma}_j = (1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1), \tag{2.5}$$

indicates a set of objects consisting only of those indexed first and last, with none of the intermediate objects included.

The fitness of any candidate is then given by the total value of that configuration of objects, $v_j = \vec{v} \cdot \vec{\gamma}_j$, but candidates are only admitted[4] if the weight of the corresponding set of objects is less than the capacity of the knapsack, i.e. $\vec{w}_j \cdot \vec{\gamma}_j \leq w_{max}$.

4 Note there are alternative strategies to dealing with candidates who violate the weight condition, such as to impose a penalty within the OF, but for our purposes let us assume we simply disregard violators.

Algorithm 1: Genetic algorithm

Input: $\mathcal{P}$ // Population of candidate models
Input: $g()$ // objective funtion
Input: map_g_to_s() // function to map fitness to selection probability
Input: select_parents() // function to select parents among generation
Input: crossover() // function to cross over two parents to produce offspring
Input: N_g // number of generations
Input: N_m // number of candidates per generation

Output: γ' // strongest candidate

$\mu \leftarrow \text{sample}\,(\mathcal{P}, N_m)$
for $i \in 1, ..., N_g$ **do**
  for $\gamma_j \in \mu$ **do**
    $g_j \leftarrow g(\gamma_j)$ // assess fitness of candidate
  end
  $\{s_j\} \leftarrow$ map_g_to_s($\{g_j\}$) // map fitnesses to normalised selection probability
  $\mu_c = \arg\max_{s_j}\{\gamma_j\}$ // record champion of this generation

  $\mu \leftarrow \{\}$ // empty set for next generation
  while $|\mu| < N_m$ **do**
    $p_1, p_2 \leftarrow$ select_parents($\{s_j\}$) // choose parents based on candidates' s_j
    $c_1, c_2 \leftarrow$ crossover(p_1, p_2) // generate offspring candidates based on parents
    for $c \in \{c_1, c_2\}$ **do**
      if $c \notin \mu$ **then**
        $\mu \leftarrow \mu \cup \{c\}$ // keep if child is new
      end
    end
  end
end
$\gamma' \leftarrow \arg\max_{s_j}\{\gamma_j \in \mu\}$ // strongest candidate on final generation

return γ'

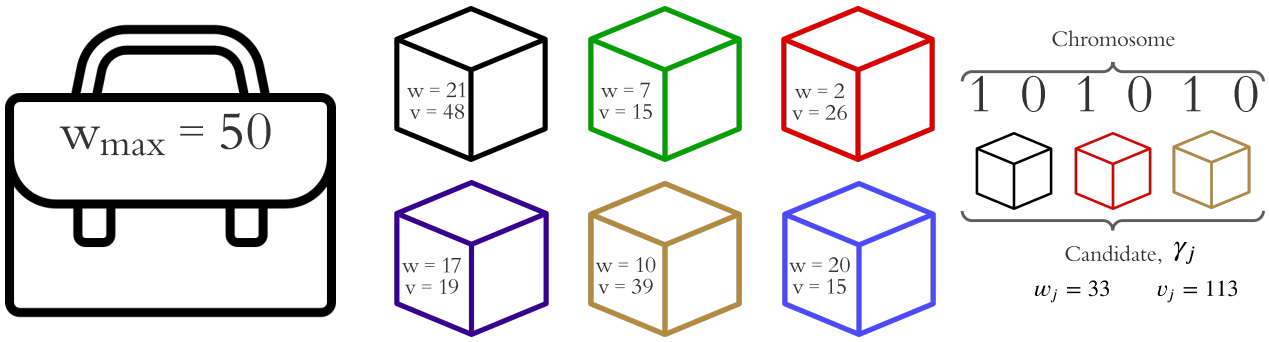

Figure 2.1: Depiction of the knapsack problem. **Left**, A knapsack which can hold any number of objects but is constrained by the total weight it can support, $w_{max} = 50$. **Centre**, A set of objects are available, each with associated weight, w, and value v. The objective is to find the subset of objects which maximise the total value, while not exceeding the capacity of the knapsack. **Right**, An example chromosome, i.e. candidate γ_j, where the bits of the chromosome indicate whether the corresponding object is included, allowing for calculation of the total weight and value of the candidate, w_j, v_j.

For example where each individual object has value < 50 and weight < 25 and $w_{max} = 50$, recalling $\gamma_j = 100001$, say,

$$\vec{v} = (48 \quad 15 \quad 26 \quad 19 \quad 39 \quad 15) \implies v_j = \vec{\gamma}_j \cdot \vec{v} = 48 + 15 = 63; \tag{2.6a}$$

$$\vec{w} = (21 \quad 7 \quad 2 \quad 17 \quad 10 \quad 20) \implies w_j = \vec{\gamma}_j \cdot \vec{w} = 21 + 20 = 41. \tag{2.6b}$$

We can hence assess the fitness of γ_j as 63 and deem it a valid candidate since it does not exceed the weight threshold. We can likewise compute the total weight and value of a series of randomly generated candidates, and deem them valid or not. Table 2.3 shows a set of 12 randomly generated candidates, of which ten are valid.

Name	Candidate	Value	Weight	Valid
γ_1	110011	117	58	No
γ_2	101010	113	33	Yes
γ_3	011110	99	36	Yes
γ_4	011011	95	39	Yes
γ_5	111000	89	30	Yes
γ_6	010111	88	54	No
γ_7	100010	87	31	Yes
γ_8	110001	78	48	Yes
γ_9	011101	75	46	Yes
γ_{10}	110000	63	28	Yes
γ_{11}	000011	54	30	Yes
γ_{12}	000101	34	37	Yes

Table 2.3: Candidate solutions to the knapsack problem for randomly generated chromosomes.

The strongest (valid) candidates from Table 2.3 are $101010, 011110$. By spawning from these candidates through a one-point crossover at the midpoint[5], we get $\gamma_{c_1} = 101110, \gamma_{c_2} = 011010$, from which we can see $v_{c_1} = 132, w_{c_1} = 50$, i.e. by combining two strong candidates we produce the strongest-yet-seen valid candidate.

By repeating this procedure, it is expected to uncover candidates which optimise v_j while maintining $w_j \leq w_{max}$, or at least to produce near-optimal solutions, using far less time/resources than brute-force evaluation of all candidates, which is usually sufficient. For instance, with $n = 100$ objects to consider, there are $2^{100} \approx 10^{30}$ candidates to consider; the most powerful supercomputers in the world currently claim on the order of Exa-FLOPs, i.e. 10^{18} operations per second, of which say $\mathcal{O}(1000)$ operations are required to test each candidate, meaning 10^{15} candidates can be checked per second in a generous example. This would still require 10^{12} seconds to solve absolutely, so it is reasonable in cases like this to accept *approximately optimal* solutions[6].

2.3.2 *Selection mechanism*

A key subroutine of every GA is the mechanism through which it nominates candidates from generation μ as parents to offsping candidates in $\mu + 1$ [90]. All mechanisms have in common that they act on a set of candidates from the previous generation, where each candidate, γ_j, has

5 One-point crossovers are detailed in Section 2.3.3 with this example shown in Fig. 2.3.

6 Simply put: in machine learning, *good enough* is good enough. We will adopt this philosophy for the remainder of this thesis and life.

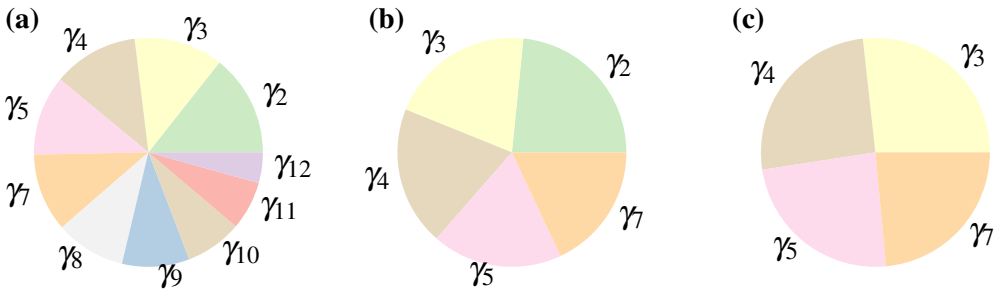

Figure 2.2: Roulette wheels showing selection probability s_i for corresponding candidates γ_i. Colours here only distinguish candidates, they do not encode any information. **b**, The set of potential parents is truncated to include only the strongest five candidates. **a**, All valid candidates are assigned selection probability based on their value in Table 2.3. **c**, After one parent (γ_2) has been chosen, it is removed from the roulette wheel and the remaining candidates' probabilities are renormalised for the selection of the second parent.

been evaluated and has fitness value, g_j. Among the viable schemes for selecting individual parents from the set of candidates, μ are

- Rank selection: candidates are selected with probabilty proportional to their ranking relative to the fitness of contemporary candidates in the same generation;
- Tournament selection: a subset of k candidates are chosen at random from μ, of which the candidate with the highest fitness is taken as the parent;
- Stochastic universal sampling: candidates are sampled proportional to their fitness, but the sampling algorithm is biased to ensure high-fitness candidates are chosen at least once within the generation.

We will only detail the mechanism used in later applications within this thesis: fitness proportional selection, known as *roulette selection* [90]. This is a straightforward strategy where we directly map candidates' fitness, g_i to a selection probability, s_i, simply by normalising $\{g_i\}$, allowing us to visualise a roulette wheel of uneven wedges, each of which correspond to a candidate. Then we need only conceptually spin the roulette wheel to select the first parent, γ_{p_1}. We then remove γ_{p_1} from the set of potential parents, renormalise the remaining $\{s_i\}$, and spin the wheel again to choose the second parent, γ_{p_2}. The roulette selection is shown in Fig. 2.2.

Practically, we repeat the process outlined until the next generation is filled, usually we have $|\mu| = N_m$, and desire that every generation should contain the same N_m candidates, so we repeat the roulette selection $N_m/2$ times per generation, since every pair of parents yield two offspring. It is important that meaningful differences in fitness are reflected by the selection probability, which is difficult to ensure for large N_m, e.g. with ten models, the strongest candidate is only a marginally more probable parent than the worst – this effect is amplified for larger N_m. We

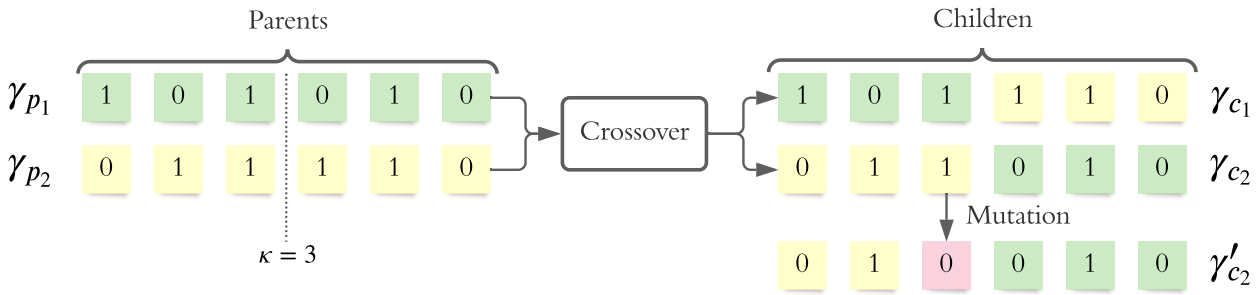

Figure 2.3: Crossover and mutation of chromosomes. Two parents, $\{\gamma_{p_1}, \gamma_{p_2}\}$, are nominated from the process in Fig. 2.2. They are then crossed-over via a one-point crossover with crossing point $\kappa = 3$, resulting in children candidates $\{\gamma_{c_1}, \gamma_{c_2}\}$. One child chromosome is mutated to yield a new candidate, γ'_{c_2}. The candidates added to the next generation are then $\{\gamma_{c_1}, \gamma'_{c_2}\}$.

therefore wish to reduce the set of potential parents to ensure high quality offspring: we truncate μ and retain only the highest-fitness $\frac{N_m}{2}$ models as selectable parents.

2.3.3 *Reproduction*

When a pair of parents have been nominated by the selection mechanism above, it remains to use those parents to *reproduce*, i.e. to produce offspring which should inherit and improve upon the properties of their parents. Here we use a *one point crossover*, whereby the two parent chromosomes are mixed together to form two offspring, about a single point, κ: for candidates of n genes, the first κ genes of γ_{p_1} are conjoined with the latter $n - \kappa$ genes of γ_{p_2}. Often κ is restricted to the midpoint of the chromosomes, although in general we need not impose this: we will instead consider $\kappa \in \left(\frac{n}{4}, \frac{3n}{4}\right)$, e.g. with $n = 12$, $\kappa \in (3, 9)$. The one-point crossover is shown for $n = 6$ with $\kappa = 3$ in Fig. 2.3, recalling the chromosome structure from Section 2.3.1.

By allowing κ other than the midpoint, we drastically increase the number of combinations of parents available for reproduction. Finally, then, parent selection is done by constructing a database of pairs of potentital parents with all available crossover points, with selection probability given by the product of their individual fitnesses. This is conceptually equivalent to selection via roulette wheel as above. Recalling the fitnesses (values) of Table 2.3, we generate the parent selection database:

Parent 1	Parent 2	κ	s_{ij}
γ_2	γ_3	2	$11,187\ (= 113 \times 99)$
γ_2	γ_3	3	$11,187$
γ_2	γ_3	4	$11,187$
γ_2	γ_4	2	$10,735\ (= 113 \times 95)$
γ_2	γ_4	3	$10,735$
γ_2	γ_4	4	$10,735$
		$\vdots$	
γ_5	γ_7	2	$7,743\ (= 89 \times 87)$
γ_5	γ_7	3	$7,743$
γ_5	γ_7	4	$7,743$

Table 2.4: Example of parent selection database. Pairs of parents are selected together, with the (unnormalised) selection probability, s_{ij}, given by the product of the individual candidates' fitnesses. Pairs of parents are repeated in the database for differing κ, and all κ are equally likely.

The GA maintains diversity in the subspace of $\mathcal{P}$ it studies, by *mutating* some of the newly proposed offspring candidates. Again, there are a multitude of approaches for this step [91], but for brevity we only describe those used in this thesis. For each proposed child candidate, γ_c, we probabilistically mutate each gene with some mutation rate r_m: if a mutation occurs, the child is replaced by γ_c'. That is, γ_c' is added to the next generation, and γ_c is discarded. r_m is a *hyperparameter* of the GA: the performance of the algorithm can be optimised by finding the best r_m for a given problem.

2.3.4 *Candidate evaluation*

Within every generation of the GA, each candidate must be evaluated, so that the relative strength of candidates can be exploited in constructing candidates for the next generation. In the example of the knapsack problem used above, candidates were evaluated by the value of their contents, but also by whether they would fit in the knapsack. Idenitfiyng the appropriate method by which to evaluate candidates is arguably the most important aspect of designing a GA: while the choice of hyperperameters (N_g, N_m, r_m) dictate the efficacy of the search, the lack of an effective metric by which to distinguish candidates would render the procedure pointless. Considerations are hence usually built into the objective function (OF); GAs implementations later in this thesis therefore demand we design OFs with respect to the individual application.

APPENDIX

A

FUNDAMENTALS

There are a number of concepts which are fundamental to any discussion of QM, but are likely to be known to most readers, and are therefore cumbersome to include in the main body of the thesis. We include them here for completeness[1].

A.1 LINEAR ALGEBRA

Here we review the language of linear algebra and summarise the basic mathematical techniques used throughout this thesis. We will briefly recall some definitions for reference.

- Notation

Definition of	Representation
Vector (or *ket*)	$\|\psi\rangle$
Dual Vector (or *bra*)	$\langle\psi\|$
Tensor Product	$\|\psi\rangle \otimes \|\phi\rangle$
Complex conjugate	$\|\psi^*\rangle$
Transpose	$\|\psi\rangle^T$
Adjoint	$\|\psi\rangle^\dagger = (\|\psi\rangle^*)^T$

Table A.1: Linear algebra definitions.

The dual vector of a vector (ket) $|\psi\rangle$ is given by $\langle\psi| = |\psi\rangle^\dagger$.

The *adjoint* of a matrix replaces each matrix element with its own complex conjugate, and then switches its columns with rows.

$$M^\dagger = \begin{pmatrix} M_{0,0} & M_{0,1} \\ M_{1,0} & M_{1,1} \end{pmatrix}^\dagger = \begin{pmatrix} M^*_{0,0} & M^*_{0,1} \\ M^*_{1,0} & M^*_{1,1} \end{pmatrix}^T = \begin{pmatrix} M^*_{0,0} & M^*_{1,0} \\ M^*_{0,1} & M^*_{1,1} \end{pmatrix} \tag{A.1}$$

1 Much of this description is reproduced from my undergraduate thesis [92].

The *inner product* of two vectors, $|\psi\rangle = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_n \end{pmatrix}$ and $|\phi\rangle = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{pmatrix}$ is given by

$$\langle\phi|\psi\rangle = (|\phi\rangle^\dagger)\,|\psi\rangle = (\phi_1^* \ \phi_2^* \ \dots \ \phi_n^*) \begin{pmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_n \end{pmatrix} = \phi_1^*\psi_1 + \phi_2^*\psi_2 + \cdots + \phi_n^*\psi_n \tag{A.2}$$

$|\psi\rangle_i, |\phi\rangle_i$ are complex numbers, and therefore the above is simply a sum of products of complex numbers. The inner product is often called the scalar product, which is in general complex.

A.2 POSTULATES OF QUANTUM MECHANICS

There are numerous statements of the postulates of quantum mechanics. Each version of the statements aims to achieve the same foundation, so we endeavour to explain them in the simplest terms.

1 Every moving particle in a conservative force field has an associated wave-function, $|\psi\rangle$. From this wave-function, it is possible to determine all physical information about the system.

2 All particles have physical properties called observables (denoted q). In order to determine a value, q, for a particular observable, there is an associated *operator* $\hat{Q}$, which, when acting on the particles wavefunction, yields the value times the wavefunction. The observable q is then the eigenvalue of the operator $\hat{Q}$.

$$\hat{Q}\,|\psi\rangle = q|\psi\rangle \tag{A.3}$$

3 Any such operator $\hat{Q}$ is Hermitian

$$\hat{Q}^\dagger = \hat{Q} \tag{A.4}$$

4 The set of eigenfunctions for any operator $\hat{Q}$ forms a complete set of linearly independent functions.

5 For a system with wavefunction $|\psi\rangle$, the expectation value of an observable q with respect to an operator $\hat{Q}$ is denoted by $\langle q\rangle$ and is given by

$$\langle q\rangle = \langle\psi|\hat{Q}|\psi\rangle \tag{A.5}$$

6 The time evolution of $|\psi\rangle$ is given by the time dependent *Schrodinger Equation*

$$i\hbar\frac{\partial\psi}{\partial t} = \hat{H}\psi, \tag{A.6}$$

where $\hat{H}$ is the system's Hamiltonian.

Using these building blocks, we can begin to construct a language to describe quantum systems.

A.3 STATES

An orhthonormal basis consists of vectors of unit length which do not overlap, e.g. $|x_1\rangle = \begin{pmatrix}1\\0\end{pmatrix}, |x_2\rangle = \begin{pmatrix}0\\1\end{pmatrix} \Rightarrow \langle x_1|x_2\rangle = 0$. In general, if $\{|x\rangle\}$ are the eigenstates of a system, then the system can be written as some state vector, $|\psi\rangle$, in general a superposition over the basis-vectors:

$$|\psi\rangle = \sum_x a_x|x\rangle \tag{A.7a}$$

$$\text{subject to} \quad \sum_x |a_x|^2 = 1, \quad a_x \in \mathbb{C} \tag{A.7b}$$

The *state space* of a physical system (classical or quantum) is then the set of all possible states the system can exist in, i.e the set of all possible values for $|\psi\rangle$ such that Eq. (A.7b) are satisfied.

For example, photons can be polarised horizontally ($\leftrightarrow$) or vertically ($\updownarrow$); take those two conditions as observable states to define the eigenstates of a two-level system, so we can designate the photon as a qubit. Then we can map the two states to a 2-dimensional, x-y plane: a general vector on such a plane can be represented by a vector with coordinates $\begin{pmatrix}x\\y\end{pmatrix}$. These polarisations can then be thought of as standard basis vectors in linear algebra. Denote $\leftrightarrow$ as the eigenstate $|0\rangle$ and $\updownarrow$ as $|1\rangle$

$$|\leftrightarrow\rangle = |0\rangle = \begin{pmatrix}1\\0\end{pmatrix} \qquad \text{A unit vector along x-axis} \tag{A.8a}$$

$$|\updownarrow\rangle = |1\rangle = \begin{pmatrix}0\\1\end{pmatrix} \qquad \text{A unit vector along y-axis} \tag{A.8b}$$

Now, in relation to the concept of superposition, we can consider, for example, a photon in an even superposition of the vertical and horizontal polarisations, evenly splitting the two basis vectors. As such, we would require that, upon measurement, it is equally likely that the

photon will *collapse* into the polarised state along x as it is to collapse along y. That is, we want $\Pr(\updownarrow) = \Pr(\leftrightarrow)$ so assign equal modulus amplitudes to the two possibilities:

$$|\psi\rangle = a\,|\leftrightarrow\rangle + b\,|\updownarrow\rangle\,, \quad \text{with} \quad \Pr(\updownarrow) = \Pr(\leftrightarrow) \Rightarrow |a|^2 = |b|^2 \tag{A.9}$$

We consider here a particular case, due to the significance of the resultant basis, where $\leftrightarrow$-polarisation and $\updownarrow$-polarisation have real amplitudes $a, b \in \mathbb{R}$.

$$\begin{aligned} &\Rightarrow a = \pm b \quad \text{but also} \quad |a|^2 + |b|^2 = 1 \\ \Rightarrow \quad &a = \frac{1}{\sqrt{2}} \quad ; \quad b = \pm\frac{1}{\sqrt{2}} \\ &\Rightarrow |\psi\rangle = \frac{1}{\sqrt{2}}\,|\leftrightarrow\rangle \pm \frac{1}{\sqrt{2}}\,|\updownarrow\rangle \\ &\Rightarrow |\psi\rangle = \frac{1}{\sqrt{2}}\,|0\rangle \pm \frac{1}{\sqrt{2}}\,|1\rangle \end{aligned} \tag{A.10}$$

These particular superpositions are of significance:

$$|+\rangle = \frac{1}{\sqrt{2}}\,(|0\rangle + |1\rangle) \tag{A.11a}$$

$$|-\rangle = \frac{1}{\sqrt{2}}\,(|0\rangle - |1\rangle) \tag{A.11b}$$

This is called the Hadamard basis: it is an equally valid vector space as the standard basis which is spanned by $\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}$, as it is simply a rotation of the standard basis.

A.3.1 *Mulitpartite systems*

In reality, we often deal with systems of multiple particles, represented by multiple qubits. Mathematically, we consider the state vector of a system containing n qubits as being the tensor product of the n qubits' individual state vectors[2]. For instance, suppose a 2-qubit system, $|\psi\rangle$ consisting of two independent qubits $|\psi_A\rangle$ and $|\psi_B\rangle$:

$$|\psi\rangle = |\psi_A\rangle\,|\psi_B\rangle = |\psi_A\psi_B\rangle = |\psi_A\rangle \otimes |\psi_B\rangle \tag{A.12}$$

Consider first a simple system of 2 qubits. Measuring in the standard basis, these qubits will have to collapse in to one of the basis states $|0,0\rangle, |0,1\rangle, |1,0\rangle, |1,1\rangle$. Thus, for such a 2-qubit system, we have the general superposition

$$|\psi\rangle = \alpha_{0,0}|0,0\rangle + \alpha_{0,1}|0,1\rangle + \alpha_{1,0}|1,0\rangle + \alpha_{1,1}|1,1\rangle$$

2 We will later discuss entangled states, which can not be described thus.

where $\alpha_{i,j}$ is the amplitude for measuring the system as the state $|i,j\rangle$. This is perfectly analogous to a classical 2-bit system necessarily occupying one of the four possibilities $\{(0,0),(0,1),(1,0),(1,1)\}$.

Hence, for example, if we wanted to concoct a two-qubit system composed of one qubit in the state $|+\rangle$ and one in $|-\rangle$

$$\begin{aligned}
|\psi\rangle &= |+\rangle \otimes |-\rangle \\
|\psi\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \otimes \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right) \\
&= \frac{1}{2}\left[|00\rangle - |01\rangle + |10\rangle - |11\rangle\right] \\
&= \frac{1}{2}\left[\begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} - \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} + \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} - \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix}\right] \\
&= \frac{1}{2}\left[\begin{pmatrix}1\\0\\0\\0\end{pmatrix} - \begin{pmatrix}0\\1\\0\\0\end{pmatrix} + \begin{pmatrix}0\\0\\1\\0\end{pmatrix} - \begin{pmatrix}0\\0\\0\\1\end{pmatrix}\right]. \\
\Rightarrow |\psi\rangle &= \frac{1}{2}\begin{pmatrix}1\\-1\\-1\\1\end{pmatrix}
\end{aligned} \tag{A.13}$$

That is, the two qubit system – and indeed any two qubit system – is given by a linear combination of the four basis vectors

$$\{|00\rangle, |0,1\rangle, |10\rangle, |11\rangle\} = \left\{\begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\1\end{pmatrix}\right\}. \tag{A.14}$$

We can notice that a single qubit system can be described by a linear combination of two basis vectors, and that a two qubit system requires four basis vectors to describe it. In general we can say that an n-qubit system is represented by a linear combination of 2^n basis vectors.

A.3.2 *Registers*

A *register* is generally the name given to an array of controllable quantum systems; here we invoke it to mean a system of multiple qubits, specifically a subset of the total number of available qubits. For example, a register of ten qubits can be denoted $|x[10]\rangle$, and we can think of the system as a register of six qubits together with a register of three and another register of one qubit.

$$|x[10]\rangle = |x_1[6]\rangle \otimes |x_2[3]\rangle \otimes |x_3[1]\rangle$$

A.4 ENTANGLEMENT

?? Another unique property of quantum systems is that of *entanglement*: when two or more particles interact in such a way that their individual quantum states can not be described independent of the other particles. A quantum state then exists for the system as a whole instead. Mathematically, we consider such entangled states as those whose state can not be expressed as a tensor product of the states of the individual qubits it's composed of: they are dependent upon the other.

To understand what we mean by this dependence, consider a counter-example. Consider the Bell state,

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right), \tag{A.15}$$

if we measure this state, we expect that it will be observed in either eigenstate $|00\rangle$ or $|11\rangle$, with equal probability due to their amplitudes' equal magnitudes. The bases for this state are simply the standard bases, $|0\rangle$ and $|1\rangle$. Thus, according to our previous definition of systems of multiple qubits, we would say this state can be given as a combination of two states, like Eq. (A.12),

$$\begin{aligned} |\Phi^+\rangle &= |\psi_1\rangle \otimes |\psi_2\rangle \\ &= (a_1|0\rangle + b_1|1\rangle) \otimes (a_2|0\rangle + b_2|1\rangle) \\ &= a_1a_2|00\rangle + a_1b_2|01\rangle + b_1a_2|10\rangle + b_1b_2|11\rangle \end{aligned} \tag{A.16}$$

However we require $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, which would imply $a_1b_2 = 0$ and $b_1a_2 = 0$. These imply that either $a_1 = 0$ or $b_2 = 0$, and also that $b_1 = 0$ or a_2=0, which are obviously invalid since we require that $a_1a_2 = b_1b_2 = \frac{1}{\sqrt{2}}$. Thus, we cannot express $|\Phi^+\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$; this inability to separate the first and second qubits is what we term *entanglement*.

A.5 UNITARY TRANSFORMATIONS

A fundamental concept in quantum mechanics is that of performing transformations on states. *Quantum transformations*, or *quantum operators*, map a quantum state into a new state within the same Hilbert space. There are certain restrictions on a physically possible quantum transformation: in order that U is a valid transformation acting on some superposition $|\psi\rangle = a_1|\psi_1\rangle + a_2|\psi_2\rangle + \ldots a_k|\psi_k\rangle$, U must be linear

$$U(a_1|\psi_1\rangle + a_2|\psi_2\rangle + \ldots a_k|\psi_k\rangle) = a_1(U|\psi_1\rangle) + a_2(U|\psi_2\rangle) + \cdots + a_k(U|\psi_k\rangle). \tag{A.17}$$

To fulfil these properties, we require that U *preserve the inner product*:

$$\langle\psi_0|U^\dagger U|\psi\rangle = \langle\psi_0|\psi\rangle$$

That is, we require that any such transformation be *unitary*:

$$UU^\dagger = I \Rightarrow U^\dagger = U^{-1} \tag{A.18}$$

Unitarity is a sufficient condition to describe any valid quantum operation: any quantum transformation can be described by a unitary transformation, and any unitary transformation corresponds to a physically implementable quantum transformation.

Then, if U_1 is a unitary transformation that acts on the space $\mathcal{H}_1$ and U_2 acts on $\mathcal{H}_2$, the product of the two unitary transformations is also unitary. The tensor product $U_1 \otimes U_2$ acts on the space $\mathcal{H}_1 \otimes \mathcal{H}_2$. So, then, supposing a system of two separable qubits, $|\psi_1\rangle$ and $|\psi_2\rangle$ where we wish to act on $|\psi_1\rangle$ with operator U_1 and on $|\psi_2\rangle$ with U_2, we perform it as

$$(U_1 \otimes U_2)\,(|\psi_1\rangle \otimes |\psi_2\rangle) = (U_1\,|\psi_1\rangle) \otimes (U_2\,|\psi_2\rangle) \tag{A.19}$$

A.6 DIRAC NOTATION

In keeping with standard practice, we employ *Dirac notation* throughout this thesis. Vectors are denoted by *kets* of the form $|a\rangle$. For example, the standard basis is represented by,

$$\begin{aligned} |x\rangle = |0\rangle &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ |y\rangle = |1\rangle &= \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{aligned} \tag{A.20}$$

We saw in Table A.1 that for every such ket, $|\psi\rangle$, there exists a *dual vector*: its complex conjugate transpose, called the *bra* of such a vector, denoted $\langle\psi|$. That is,

$$\begin{aligned} \langle\psi|^\dagger &= |\psi\rangle \\ |\psi\rangle^\dagger &= \langle\psi| \end{aligned} \tag{A.21}$$

$$|\psi\rangle = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_n \end{pmatrix} \Rightarrow \langle\psi| = \begin{pmatrix} \psi_1^* & \psi_2^* & \dots & \psi_n^* \end{pmatrix} \tag{A.22}$$

Then if we have two vectors $|\psi\rangle$ and $|\phi\rangle$, their *inner product* is given as $\langle\psi|\phi\rangle = \langle\phi|\psi\rangle$.

$$\begin{aligned} |\psi\rangle = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \vdots \\ \psi_n \end{pmatrix} \quad ; \quad |\phi\rangle = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_n \end{pmatrix} \\ \Rightarrow \langle\phi| = \begin{pmatrix} \phi_1^* & \phi_2^* & \phi_3^* & \cdots & \phi_n^* \end{pmatrix} \\ \Rightarrow \langle\phi|\,|\psi\rangle = \begin{pmatrix} \phi_1^* & \phi_2^* & \phi_3^* & \cdots & \phi_n^* \end{pmatrix} \begin{pmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \vdots \\ \psi_n \end{pmatrix} \\ \Rightarrow \langle\phi|\,|\psi\rangle = \phi_1^*\psi_1 + \phi_2^*\psi_2 + \phi_3^*\psi_3 + \cdots + \phi_n^*\psi_n \end{aligned} \tag{A.23}$$

Example A.6.1.

$$\begin{aligned} |\psi\rangle = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \quad ; \quad |\phi\rangle = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} \\ \Rightarrow \langle\phi|\,|\psi\rangle = \begin{pmatrix} 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \\ = (4)(1) + (5)(2) + (6)(3) = 32 \end{aligned} \tag{A.24}$$

Similarly, their *outer product* is given as $|\phi\rangle\langle\psi|$. Multiplying a column vector by a row vector thus gives a matrix. Matrices generated by a outer products then define operators:

Example A.6.2.

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} \begin{pmatrix} 3 & 4 \end{pmatrix} = \begin{pmatrix} 3 & 4 \\ 6 & 8 \end{pmatrix} \tag{A.25}$$

Then we can say, for $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

$$|0\rangle\langle 0| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \tag{A.26a}$$

$$|0\rangle\langle 1| = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \tag{A.26b}$$

$$|1\rangle\langle 0| = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \tag{A.26c}$$

$$|1\rangle\langle 1| = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{A.26d}$$

And so any 2-dimensional linear transformation in the standard basis $|0\rangle, |1\rangle$ can be given as a sum

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} = a|0\rangle\langle 0| + b|0\rangle\langle 1| + c|1\rangle\langle 0| + d|1\rangle\langle 1| \tag{A.27}$$

This is a common method of representing operators as outer products of vectors. A transformation that *exchanges* a particle between two states, say $|0\rangle \leftrightarrow |1\rangle$ is given by the operation

$$\hat{Q} : \begin{cases} |0\rangle \to |1\rangle \\ |1\rangle \to |0\rangle \end{cases}$$

Which is equivalent to the outer product representation

$$\hat{Q} = |0\rangle\langle 1| + |1\rangle\langle 0|$$

For clarity, here we will prove this operation

Example A.6.3.

$$\begin{aligned} \hat{Q} &= |0\rangle\langle 1| + |1\rangle\langle 0| \\ &= \begin{pmatrix} 1 \\ 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \end{aligned}$$

So then, acting on $|0\rangle$ *and* $|1\rangle$ *gives*

$$\hat{Q}|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$

$$\hat{Q}|1\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle$$

To demonstrate how Dirac notation simplifies this:

$$\begin{aligned}\hat{Q}\left|0\right\rangle &= \left(\left|0\right\rangle\left\langle 1\right| + \left|1\right\rangle\left\langle 0\right|\right)\left|0\right\rangle \\ &= \left|0\right\rangle\left\langle 1\right|\left|0\right\rangle + \left|1\right\rangle\left\langle 0|0\right\rangle \\ &= \left|0\right\rangle\left\langle 1|0\right\rangle + \left|1\right\rangle\left\langle 0|0\right\rangle\end{aligned}$$

Then, since $|0\rangle$ and $|1\rangle$ are orthogonal basis, their inner product is 0 and the inner product of a vector with itself is 1, ($\langle 1|1\rangle = \langle 0|0\rangle = 1, \langle 0|1\rangle = \langle 1|0\rangle = 0$). So,

$$\begin{aligned}\hat{Q}\left|0\right\rangle &= \left|0\right\rangle(0) + \left|1\right\rangle(1) \\ \Rightarrow \hat{Q}\left|0\right\rangle &= \left|1\right\rangle\end{aligned} \tag{A.28}$$

And similarly for $\hat{Q}\left|1\right\rangle$. This simple example then shows why Dirac notation can significantly simplify calculations across quantum mechanics, compared to standard matrix and vector notation. To see this more clearly, we will examine a simple 2-qubit state under such operations. The method generalises to operating on two or more qubits generically: we can define any operator which acts on two qubits as a sum of outer products of the basis vectors $\left|00\right\rangle, \left|01\right\rangle, \left|10\right\rangle$ and $\left|11\right\rangle$. We can similarly define any operator which acts on an n qubit state as a linear combination of the 2^n basis states generated by the n qubits.

Example A.6.4. *To define a transformation that will exchange basis vectors $|00\rangle$ and $|11\rangle$, while leaving $|01\rangle$ and $|10\rangle$ unchanged (ie exchanging $|01\rangle \leftrightarrow |01\rangle, |10\rangle \leftrightarrow |10\rangle$) we define an operator*

$$\hat{Q} = \left|00\right\rangle\left\langle 11\right| + \left|11\right\rangle\left\langle 00\right| + \left|10\right\rangle\left\langle 10\right| + \left|01\right\rangle\left\langle 01\right| \tag{A.29}$$

Then, using matrix calculations this would require separately calculating the four outer products in the above sum and adding them to find a 4×4 matrix to represent $\hat{Q}$, which then acts on a state $|\psi\rangle$. Instead, consider first that $|\psi\rangle = |00\rangle$, ie one of the basis vectors our transformation is to change:

$$\hat{Q}\left|00\right\rangle = \left(\left|00\right\rangle\left\langle 11\right| + \left|11\right\rangle\left\langle 00\right| + \left|10\right\rangle\left\langle 10\right| + \left|01\right\rangle\left\langle 01\right|\right)\ \left|00\right\rangle \tag{A.30}$$

And as before, only the inner products of a vector with itself remains:

$$\begin{aligned}&= \left|00\right\rangle\left\langle 11\left|00\right.\right\rangle + \left|11\right\rangle\left\langle 00\left|00\right.\right\rangle + \left|10\right\rangle\left\langle 10\left|00\right.\right\rangle + \left|01\right\rangle\left\langle 01\left|00\right.\right\rangle \\ &= \left|00\right\rangle(0) + \left|11\right\rangle(1) + \left|10\right\rangle(0) + \left|01\right\rangle(0) \\ \Rightarrow &\hat{Q}\left|00\right\rangle = \left|11\right\rangle\end{aligned} \tag{A.31}$$

i.e the transformation has performed $\hat{Q} : |00\rangle \rightarrow |11\rangle$ as expected. Then, if we apply the same transformation to a state which does not depend on one of the target states, eg,

$$\begin{aligned}\left|\psi\right\rangle =& a\left|10\right\rangle + b\left|01\right\rangle \\ \hat{Q}\left|\psi\right\rangle =& \Big(\left|00\right\rangle\left\langle 11\right| + \left|11\right\rangle\left\langle 00\right| + \left|10\right\rangle\left\langle 10\right| + \left|01\right\rangle\left\langle 01\right|\Big)\Big(a\left|10\right\rangle + b\left|01\right\rangle\Big) \\ =& a\Big(\left|00\right\rangle\left\langle 11\left|10\right.\right\rangle + \left|11\right\rangle\left\langle 00\left|10\right.\right\rangle + \left|10\right\rangle\left\langle 10\left|10\right.\right\rangle + \left|01\right\rangle\left\langle 01\left|10\right.\right\rangle\Big) \\ &+ b\Big(\left|00\right\rangle\left\langle 11\left|01\right.\right\rangle + \left|11\right\rangle\left\langle 00\left|01\right.\right\rangle + \left|10\right\rangle\left\langle 10\left|01\right.\right\rangle + \left|01\right\rangle\left\langle 01\left|01\right.\right\rangle\Big)\end{aligned} \tag{A.32}$$

And since the inner product is a scalar, we can factor terms such as $\langle 11|10\rangle$ *to the beginning of expressions, eg* $|00\rangle \langle 11| |10\rangle = \langle 11 |10\rangle |00\rangle$*, and we also know*

$$\begin{aligned} \langle 11|10\rangle = \langle 00|10\rangle = \langle 01|10\rangle = \langle 11|01\rangle = \langle 00|01\rangle = \langle 10|01\rangle = 0 \\ \langle 10|10\rangle = \langle 01|01\rangle = 1 \end{aligned} \tag{A.33}$$

We can express the above as

$$\begin{aligned} \hat{Q}\,|\psi\rangle = a\Big((0)\,|00\rangle + (0)\,|11\rangle + (1)\,|10\rangle + (0)\,|01\rangle\Big) \\ +b\Big((0)\,|00\rangle + (0)\,|11\rangle + (0)\,|10\rangle + (1)|01\rangle\Big) \\ =a\,|10\rangle\,| + b|01\rangle \\ =|\psi\rangle \end{aligned} \tag{A.34}$$

Then it is clear that, when $|\psi\rangle$ *is a superposition of states unaffected by transformation* $\hat{Q}$*, then* $\hat{Q}\,|\psi\rangle = |\psi\rangle$*.*

This method generalises to systems with greater numbers of particles (qubits). If we briefly consider a 3 qubit system - and initialise all qubits in the standard basis state $|0\rangle$ - then the system is represented by $|000\rangle = |0\rangle \otimes |0\rangle \otimes |0\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix}$. This quantity is an 8-row vector. To calculate the outer product $\langle 000|000\rangle$, we would be multiplying an 8-column bra $\langle 000|$ by an 8-row ket $|000\rangle$. Clearly then we will be working with 8×8 matrices, which will become quite difficult to maintain effectively and efficiently quite fast. As we move to systems of larger size, standard matrix multiplication becomes impractical for hand-written analysis, although of course remains tractable computationally up to $n \sim 10$ qubits. It is obvious that Dirac's bra/ket notation is a helpful, pathematically precise tool for QM.

BIBLIOGRAPHY

[1] Max Jammer et al. *The conceptual development of quantum mechanics*. McGraw-Hill New York, 1966.

[2] Albert Einstein. On a heuristic point of view concerning the production and transformation of light. *Annalen der Physik*, pages 1–18, 1905.

[3] Max Born. Quantenmechanik der stoßvorgänge. *Zeitschrift für Physik*, 38(11-12):803–827, 1926.

[4] Erwin Schrödinger. An undulatory theory of the mechanics of atoms and molecules. *Physical review*, 28(6):1049, 1926.

[5] Werner Heisenberg. Über quantentheoretische umdeutung kinematischer und mechanischer beziehungen. In *Original Scientific Papers Wissenschaftliche Originalarbeiten*, pages 382–396. Springer, 1985.

[6] John Von Neumann. *Mathematical foundations of quantum mechanics: New edition*. Princeton university press, 2018.

[7] Orazio Svelto and David C Hanna. *Principles of lasers*, volume 1. Springer, 2010.

[8] Bart Van Zeghbroeck. Principles of semiconductor devices. *Colarado University*, 34, 2004.

[9] David J Griffiths and Darrell F Schroeter. *Introduction to quantum mechanics*. Cambridge University Press, 2018.

[10] Leonard Susskind and Art Friedman. *Quantum mechanics: the theoretical minimum*. Basic Books, 2014.

[11] Eleanor G Rieffel and Wolfgang H Polak. *Quantum computing: A gentle introduction*. MIT Press, 2011.

[12] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.

[13] Paul Adrien Maurice Dirac. *The principles of quantum mechanics*. Number 27. Oxford university press, 1981.

[14] T Mart. How do i introduce schr\" odinger equation during the quantum mechanics course? *arXiv preprint arXiv:2010.15589*, 2020.

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

[16] Leonard Susskind and George Hrabovsky. *Classical mechanics: the theoretical minimum*. Penguin Books, 2014.

[17] Heinz-Peter Breuer, Francesco Petruccione, et al. *The theory of open quantum systems*. Oxford University Press on Demand, 2002.

[18] Daniel Manzano. A short introduction to the lindblad master equation. *AIP Advances*, 10(2):025106, 2020.

[19] John Preskill. Lecture notes for physics 229: Quantum information and computation. *California Institute of Technology*, 16, 1998.

[20] Jonathan P Dowling and Gerard J Milburn. Quantum technology: the second quantum revolution. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 361(1809):1655–1674, 2003.

[21] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum-enhanced measurements: beating the standard quantum limit. *Science*, 306(5700):1330–1336, 2004.

[22] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Advances in quantum metrology. *Nature photonics*, 5(4):222, 2011.

[23] Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *arXiv preprint arXiv:2003.06557*, 2020.

[24] Artur K Ekert. Quantum cryptography based on bell's theorem. *Physical review letters*, 67(6):661, 1991.

[25] Nicolas Gisin, Grégoire Ribordy, Wolfgang Tittel, and Hugo Zbinden. Quantum cryptography. *Reviews of modern physics*, 74(1):145, 2002.

[26] Yu I Manin. Vychislimoe i nevychislimoe (computable and noncomputable), moscow: Sov, 1980.

[27] Paul 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(5):563–591, 1980.

[28] Paul Benioff. Quantum mechanical hamiltonian models of turing machines. *Journal of Statistical Physics*, 29(3):515–546, 1982.

[29] Richard P Feynman. Simulating physics with computers. *Int. J. Theor. Phys*, 21(6/7), 1982.

[30] Seth Lloyd. Universal quantum simulators. *Science*, pages 1073–1078, 1996.

[31] J Ignacio Cirac and Peter Zoller. Goals and opportunities in quantum simulation. *Nature Physics*, 8(4):264–266, 2012.

[32] Benjamin P Lanyon, James D Whitfield, Geoff G Gillett, Michael E Goggin, Marcelo P Almeida, Ivan Kassal, Jacob D Biamonte, Masoud Mohseni, Ben J Powell, Marco Barbieri, et al. Towards quantum chemistry on a quantum computer. *Nature chemistry*, 2(2):106–111, 2010.

[33] Yudong Cao, Jonathan Romero, Jonathan P Olson, Matthias Degroote, Peter D Johnson, Mária Kieferová, Ian D Kivlichan, Tim Menke, Borja Peropadre, Nicolas PD Sawaya, et al. Quantum chemistry in the age of quantum computing. *Chemical reviews*, 119(19):10856–10915, 2019.

[34] Sam McArdle, Suguru Endo, Alan Aspuru-Guzik, Simon C Benjamin, and Xiao Yuan. Quantum computational chemistry. *Reviews of Modern Physics*, 92(1):015003, 2020.

[35] Alán Aspuru-Guzik, Anthony D Dutoi, Peter J Love, and Martin Head-Gordon. Simulated quantum computation of molecular energies. *Science*, 309(5741):1704–1707, 2005.

[36] Chris Sparrow, Enrique Martín-López, Nicola Maraviglia, Alex Neville, Christopher Harrold, Jacques Carolan, Yogesh N Joglekar, Toshikazu Hashimoto, Nobuyuki Matsuda, Jeremy L O'Brien, et al. Simulating the vibrational quantum dynamics of molecules using photonics. *Nature*, 557(7707):660–667, 2018.

[37] David Deutsch. Quantum theory, the church–turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400(1818):97–117, 1985.

[38] Daniel R Simon. On the power of quantum computation. *SIAM journal on computing*, 26(5):1474–1483, 1997.

[39] Charles H Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523, 1997.

[40] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on computing*, 26(5):1411–1473, 1997.

[41] Lov K Grover. Quantum mechanics helps in searching for a needle in a haystack. *Physical review letters*, 79(2):325, 1997.

[42] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[43] John Watrous. Quantum computational complexity. *arXiv preprint arXiv:0804.3401*, 2008.

[44] Peter W Shor. Fault-tolerant quantum computation. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 56–65. IEEE, 1996.

[45] Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error rate. *SIAM Journal on Computing*, 2008.

[46] David P DiVincenzo. The physical implementation of quantum computation. *Fortschritte der Physik: Progress of Physics*, 48(9-11):771–783, 2000.

[47] Travis S Humble, Himanshu Thapliyal, Edgard Munoz-Coreas, Fahd A Mohiyaddin, and Ryan S Bennink. Quantum computing circuits and devices. *IEEE Design & Test*, 36(3):69–94, 2019.

[48] Pieter Kok, William J Munro, Kae Nemoto, Timothy C Ralph, Jonathan P Dowling, and Gerard J Milburn. Linear optical quantum computing with photonic qubits. *Reviews of Modern Physics*, 79(1):135, 2007.

[49] Jeremy C Adcock, Jueming Bao, Yulin Chi, Xiaojiong Chen, Davide Bacco, Qihuang Gong, Leif K Oxenløwe, Jianwei Wang, and Yunhong Ding. Advances in silicon quantum photonics. *IEEE Journal of Selected Topics in Quantum Electronics*, 27(2):1–24, 2020.

[50] John Michael Kovac, EM Leitch, C Pryke, JE Carlstrom, NW Halverson, and WL Holzapfel. Detection of polarization in the cosmic microwave background using dasi. *Nature*, 420(6917):772–787, 2002.

[51] Christopher Gerry, Peter Knight, and Peter L Knight. *Introductory quantum optics*. Cambridge university press, 2005.

[52] Robert Raussendorf, Daniel E Browne, and Hans J Briegel. Measurement-based quantum computation on cluster states. *Physical review A*, 68(2):022312, 2003.

[53] Caterina Vigliar, Stefano Paesani, Yunhong Ding, Jeremy C Adcock, Jianwei Wang, Sam Morley-Short, Davide Bacco, Leif K Oxenløwe, Mark G Thompson, John G Rarity, et al. Error protected qubits in a silicon photonic chip. *arXiv preprint arXiv:2009.08339*, 2020.

[54] Stefano Paesani, Massimo Borghi, Stefano Signorini, Alexandre Maïnos, Lorenzo Pavesi, and Anthony Laing. Near-ideal spontaneous photon sources in silicon quantum photonics. *Nature communications*, 11(1):1–6, 2020.

[55] Michel H Devoret, Andreas Wallraff, and John M Martinis. Superconducting qubits: A short review. *arXiv preprint cond-mat/0411174*, 2004.

[56] Morten Kjaergaard, Mollie E Schwartz, Jochen Braumüller, Philip Krantz, Joel I-J Wang, Simon Gustavsson, and William D Oliver. Superconducting qubits: Current state of play. *Annual Review of Condensed Matter Physics*, 11:369–395, 2020.

[57] M Kjaergaard, ME Schwartz, A Greene, GO Samach, A Bengtsson, M O'Keeffe, CM McNally, J Braumüller, DK Kim, P Krantz, et al. A quantum instruction set implemented on a superconducting quantum processor. *arXiv preprint arXiv:2001.08838*, 2020.

[58] John M Martinis and A Megrant. Ucsb final report for the csq program: Review of decoherence and materials physics for superconducting qubits. *arXiv preprint arXiv:1410.5793*, 2014.

[59] Göran Wendin. Quantum information processing with superconducting circuits: a review. *Reports on Progress in Physics*, 80(10):20, 2017.

[60] Ioan M Pop, Kurtis Geerlings, Gianluigi Catelani, Robert J Schoelkopf, Leonid I Glazman, and Michel H Devoret. Coherent suppression of electromagnetic dissipation due to superconducting quasiparticles. *Nature*, 508(7496):369–372, 2014.

[61] Jay M Gambetta, Jerry M Chow, and Matthias Steffen. Building logical qubits in a superconducting quantum computing system. *npj Quantum Information*, 3(1):1–7, 2017.

[62] David Kielpinski, Chris Monroe, and David J Wineland. Architecture for a large-scale ion-trap quantum computer. *Nature*, 417(6890):709–711, 2002.

[63] Christopher Monroe and Jungsang Kim. Scaling the ion trap quantum processor. *Science*, 339(6124):1164–1169, 2013.

[64] John P Gaebler, Ting Rei Tan, Y Lin, Y Wan, R Bowler, Adam C Keith, S Glancy, K Coakley, E Knill, D Leibfried, et al. High-fidelity universal gate set for be 9+ ion qubits. *Physical review letters*, 117(6):060505, 2016.

[65] Ye Wang, Mark Um, Junhua Zhang, Shuoming An, Ming Lyu, Jing-Ning Zhang, L-M Duan, Dahyun Yum, and Kihwan Kim. Single-qubit quantum memory exceeding ten-minute coherence time. *Nature Photonics*, 11(10):646–650, 2017.

[66] AH Myerson, DJ Szwer, SC Webster, DTC Allcock, MJ Curtis, G Imreh, JA Sherman, DN Stacey, AM Steane, and DM Lucas. High-fidelity readout of trapped-ion qubits. *Physical Review Letters*, 100(20):200502, 2008.

[67] VM Schäfer, CJ Ballance, K Thirumalai, LJ Stephenson, TG Ballance, AM Steane, and DM Lucas. Fast quantum logic gates with trapped-ion qubits. *Nature*, 555(7694):75–78, 2018.

[68] Colin D Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M Sage. Trapped-ion quantum computing: Progress and challenges. *Applied Physics Reviews*, 6(2):021314, 2019.

[69] Ryan LaRose. Overview and comparison of gate level quantum software platforms. *Quantum*, 3:130, 2019.

[70] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

[71] Aram W Harrow and Ashley Montanaro. Quantum computational supremacy. *Nature*, 549(7671):203–209, 2017.

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

[73] Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, et al. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, 2020.

[74] The Difference Between AI and Machine Learning, Jan 2021. [Online; accessed 7. Jan. 2021].

[75] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[76] Alan M Turing. Computing machinery and intelligence. In *Parsing the turing test*, pages 23–65. Springer, 2009.

[77] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 2002.

[78] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[79] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

[80] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[81] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[82] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168, 2006.

[83] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24, 2007.

[84] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[85] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[86] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[87] Vedran Dunjko and Hans J Briegel. Machine learning & artificial intelligence in the quantum domain: a review of recent progress. *Reports on Progress in Physics*, 81(7):074001, 2018.

[88] Thomas Back. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996.

[89] Kenneth De Jong. Evolutionary computation: a unified approach. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pages 327–342, 2020.

[90] Sean Luke. 1 essentials of metaheuristicsl. 1.

[91] Lothar M Schmitt. Theory of genetic algorithms. *Theoretical Computer Science*, 259(1-2):1–61, 2001.

[92] Brian Flynn. Mathematical introduction to quantum computation, 2015. Undergraduate thesis.