Quantum Algorithms through Linear Algebra

Lecture Notes for CSE 410

Matthew G. Knepley

Department of Computer Science and Engineering
University At Buffalo

October 17, 2022

I dedicate these notes to my wonderful wife Margarete, without whose patience and help they could never have been written.

Acknowledgements

TBD

Contents

Chapter 1

Introduction

In this course, we will be mainly concerned with discrete binary systems, meaning those for which a measurement returns a 0 or 1, or true/false, heads/tails, up/down, Hall/Oates, or any other dichotomy you wish to use. It is definitely possible to use ternary systems, or k-ary, but almost all physical models of quantum computing use binary. We will write the state of such a system in at least two ways. First, we have the familiar linear-algebraic notation for a two-state system,

$$\text{down} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad \text{up} = \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

where each possible state is assigned a basis vector. Another popular notation, known as *Dirac notation*, names the basis vectors explicitly

$$\text{down} = |0\rangle \qquad \text{up} = |1\rangle\,,$$

or even

$$\text{down} = |d\rangle \qquad \text{up} = |u\rangle\,.$$

We could make our linear algebra look more like Dirac notation by using basis vectors $\hat{\mathbf{e}}_i$ explicitly

$$\text{down} = \hat{\mathbf{e}}_0 \qquad \text{up} = \hat{\mathbf{e}}_1.$$

We will call our two-state system a *bit*, which is a portmanteau of "binary digit". Claude E. Shannon first used the word bit in his seminal 1948 paper, A Mathematical Theory of Communication (Shannon 1948), and attributed its origin to John W. Tukey. Many times it will be helpful to think of an actual system, such as a coin. If the coin shows tails, we will say that it is in state $\hat{\mathbf{e}}_0$ or $|T\rangle$, but for heads it is in state $\hat{\mathbf{e}}_1$ or $|H\rangle$. For a normal, deterministic coin in order to specify the state, we merely give the appropriate basis vector, heads or tails. However, this is not the right setting for understanding the quantum analogue.

Instead, let us imagine flipping the coin. Now we have introduced indeterminacy, or stochasticity, into our system. What can we say about the state of our coin after flipping, before we lift our hand? We would probably say that it is equally likely to be heads or tails. In our notation above, we could write

$$\begin{aligned}\frac{1}{2}\text{down} + \frac{1}{2}\text{up} &= \frac{1}{2}|T\rangle + \frac{1}{2}|H\rangle \\ &= \frac{1}{2}\hat{\mathbf{e}}_0 + \frac{1}{2}\hat{\mathbf{e}}_1\end{aligned}$$

where we interpret the coefficient in front of each basis vector as "the chance our system ends up in this state". This can be thought of now as a probabilistic bit, or *probit*. The chance of observing a certain outcome $|j\rangle$ from state $|\psi\rangle$ is then $\langle j|\psi\rangle$, or using linear algebra $\hat{\mathbf{e}}_j^\dagger \boldsymbol{\psi}$. This is very close to the result for a quantum mechanical system, for which the chance of observation is the square of this quantity. We will see in later chapters that the proper classical analogues to quantum mechanical systems are probabilistic, not deterministic, classical systems.

Now suppose we want to change the probit system by changing the probabilities of getting either state, from $(\frac{1}{2}\,\frac{1}{2})$ to $(\,p_1\;p_2\,)$. Since the two states are the only possible outcomes, we would still want the sum of the two coefficients to be one, meaning that we are guaranteed to get one of them. If, in addition, we demand that the change be linear. we would have a matrix equation

$$\begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}$$

where the matrix U has nonnegative entries, and its columns must sum to one, which is called a *left stochastic matrix*. We note that this matrix preserves the 1-norm of the input vector $\vec{\mathbf{p}}$. We will see that quantum evolution uses unitary matrices that preserve the 2-norm of the input vectors of quantum amplitudes.

We can create a more abstract setting for the ideas we have discussed above. Let us call a *state* a description of our system which is sufficient to predict the outcome of any measurement, and the number of real parameters (or measurements) necessary to define the state will be the number of *degrees of freedom* K. The *dimension* of our system will be the maximum number of states which can be distinguished by a single measurement. For example, our single coin system has $K = 2$ degrees of freedom, p_0 and p_1, as well as dimension two, $p\,|0\rangle$ and $(1-p)\,|1\rangle$. Note that the dimension is also the number of basis vectors, matching the usual definition from linear algebra. We will call the *probability* of an event, the relative frequency of its occurrence when the measurement is performed on a ensemble of n identically prepared systems in the limit as n becomes infinite.

The important thing to explore is how composite systems behave, namely those that are composed of collections of simpler systems. Suppose that we have two coins. Then the possible configurations for this system are

$$|TT\rangle, |TH\rangle, |HT\rangle, |HH\rangle$$

so that it has dimension four. We will be able to specify the outcomes of measurements using four probability weights $(\,p_1\ p_2\ p_3\ p_4\,)$, so that $K = N$. We expect that if we fix one of the coins, then this system will behave just like a one probit system with a single coin, which is indeed that case. When we combined the two coins, we saw that $N_2 = N_1 \cdot N_1$ and likewise $K_2 = K_1 \cdot K_1$. We will take this as a general axiom for any two systems. Given these assumptions about the behavior of composite systems, one can prove (Hardy 2001; Schack 2003) that

$$K = N^r$$

where r is a positive integer. We will define a *pure state* as a state that is the result of a measurement. For example, if I look at the result of a coin toss, I can only see a head or a tail, not some combination. Thus, even though the state of my classical probabilistic system can be expressed as an infinite number of vectors $\alpha\,|H\rangle + (1-\alpha)\,|T\rangle$, there are only two pure states. In quantum mechanics, however, I can measure a superposition of states. For example, I can measure light polarization at any angle even though there are only two basis vectors, say horizontal and vertical polarization. Thus there are an infinity of pure states.

If one insists that a reversible, continuous transformation exist between the pure states of the system, we can rule out $r = 1$ since there are a finite number of pure states in this case. This situation is exactly classical probability theory, and the pure states correspond to the basis vectors. In quantum theory, we have an infinity of of pure states so that a continuous transformation between them is possible, and a full state is described by a *density matrix* which has N^2 real parameters.

1.1 The Quantum Mechanical Setting

Complex Hilbert space is the setting for quantum mechanics. Hardy (Hardy 2001) shows that this is related to the behavior of composite systems. In real Hilbert space, composite systems have too many degrees of freedom, and in quaternionic Hilbert space they have too few. The signature operation in a Hilbert space, the inner product $\langle\phi|\psi\rangle$, is defined by three properties:

1. **Positivity**: $\langle\psi|\psi\rangle > 0 \qquad \forall\,\psi \neq 0.$
2. **Conjugate Symmetry**: $\langle\phi|\psi\rangle = \overline{\langle\psi|\phi\rangle}$
3. **Linearity**: $\langle\phi|\,(a\,|\psi\rangle + b\,|\omega\rangle) = a\,\langle\phi|\psi\rangle + b\,\langle\phi|\omega\rangle$

The complex Hilbert space $\mathcal{H}$ is complete in the norm induced by the inner product

$$\|\psi\|^2 = \langle\psi|\psi\rangle \tag{1.1}$$

With this space, we can now define the mathematical objects representing our physical objects. First we will state the common definitions. These are quite clean, but misleading, since they only represent isolated systems. We will see afterwards that for composite systems a more complicated approach is necessary.

States A *state* of our physical system, meaning the information sufficient to determine the probabilistic outcome of any measurement. This means that knowing the state and being able to prepare many identical systems in this state, I can predict the probability (long run average) of any measurement. Our state will be a *ray* in $\mathcal{H}$, meaning the equivalence class of vectors $a\psi$ or any scaling $a \in \mathbb{C}$, including scalings $e^{i\alpha}$ which preserve the norm, called a *phase*.

Observables An *observable* is a property of a physical system that can be measured. In quantum mechanics, an observable is a self-adjoint linear operator. The eigenstates of a self-adjoint linear operator form an orthonormal basis, and therefore A has a *spectral representation* in terms of these states ϕ_i

$$A = \sum_i \lambda_i \left|\phi_i\right\rangle\!\left\langle\phi_i\right| = \sum_i \lambda_i P_i, \tag{1.2}$$

where P_i is the orthogonal projector onto the ith eigenspace.

Measurement When we measure A for a quantum state ψ, this collapses the system into an eigenstate ϕ of A and the value of the measurement is the eigenvalue λ. We will define a *measurement* of an eigenstate ϕ of observable A on the state ψ to be a map M from $A\left|\psi\right\rangle$ to the real numbers

$$M : \mathcal{H} \to \mathbb{R}$$

which gives the probability of obtaining state ϕ and value λ after the operation. This probability is given by

$$\Pr(\lambda) = \left\|P\left|\psi\right\rangle\right\|^2 = \left\langle\psi|P|\psi\right\rangle \tag{1.3}$$

where we have used the fact that P is a projector.

Now we consider a composite system AB composed of two subsystems A and B. The state ψ_{AB} of this composite system lives in the Hilbert space $\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B$. For the next example, we will consider a two qubit state, but we can easily generalize this to bigger systems. If we want to measure observable L_A only for the A qubit, our combined observable L will be

$$L = L_A \otimes I$$

where I is the identity operator on $\mathcal{H}_B$. If we look at the expectation value of L,

$$\begin{aligned}
\left\langle\psi|L|\psi\right\rangle = &\left(\bar{a}\left\langle 0\right| \otimes \left\langle 0\right| + \bar{b}\left\langle 0\right| \otimes \left\langle 1\right| + \bar{c}\left\langle 1\right| \otimes \left\langle 0\right| + \bar{d}\left\langle 1\right| \otimes \left\langle 1\right|\right)(L_A \otimes I) \\
&\left(a\left|0\right\rangle \otimes \left|0\right\rangle + b\left|0\right\rangle \otimes \left|1\right\rangle + c\left|1\right\rangle \otimes \left|0\right\rangle + d\left|1\right\rangle \otimes \left|1\right\rangle\right) \qquad (1.4) \\
= &\left|a\right|^2\left\langle 0|L_A|0\right\rangle + \bar{a}c\left\langle 0|L_A|1\right\rangle + \left|b\right|^2\left\langle 0|L_A|0\right\rangle + \bar{b}d\left\langle 0|L_A|1\right\rangle \\
&+ \left|c\right|^2\left\langle 1|L_A|1\right\rangle + \bar{c}a\left\langle 1|L_A|0\right\rangle + \left|d\right|^2\left\langle 1|L_A|1\right\rangle + \bar{d}b\left\langle 1|L_A|0\right\rangle \qquad (1.5) \\
= &\left(\left|a\right|^2 + \left|b\right|^2\right)\left\langle 0|L_A|0\right\rangle + 2\,\mathrm{Re}\left\{\bar{a}c + \bar{b}d\right\}\left\langle 0|L_A|1\right\rangle \\
&+ \left(\left|c\right|^2 + \left|d\right|^2\right)\left\langle 1|L_A|1\right\rangle \qquad (1.6)
\end{aligned}$$

This expression can be rewritten into a matrix equation

$$\begin{aligned}\langle\psi|L|\psi\rangle &= \left(|a|^2+|b|^2\right)\mathrm{Tr}\{L_A\,|0\rangle\langle 0|\}+\left(|c|^2+|d|^2\right)\mathrm{Tr}\{L_A\,|1\rangle\langle 1|\}\\ &\quad+\mathrm{Re}\{\bar{a}c+\bar{b}d\}\,\mathrm{Tr}\{L_A\,|0\rangle\langle 1|\}+\mathrm{Re}\{\bar{a}c+\bar{b}d\}\,\mathrm{Tr}\{L_A\,|1\rangle\langle 0|\} &(1.7)\\ &= \mathrm{Tr}\left\{L_A\begin{pmatrix}|a|^2+|b|^2 & \mathrm{Re}\{\bar{a}c+\bar{b}d\}\\ \mathrm{Re}\{\bar{a}c+\bar{b}d\} & |c|^2+|d|^2\end{pmatrix}\right\} &(1.8)\\ &= \mathrm{Tr}\{L_A\rho_A\} &(1.9)\end{aligned}$$

where ρ_A is called the *density operator* for system A, and multiplication by L_A is component-wise, not matrix multiplication. Clearly the density operator is self-adjoint, and has real entries. It also has trace 1 since the initial combined state was normalized, and it has only positive eigenvalues (which might not be immediately clear).

Since this form for the expectation value of L is true for any observable acting on system A, we can interpret ρ_A as defining a statistical ensemble of quantum states for system A rather than a set of states linked to the states of system B. For example, suppose we have the simple state

$$a\,|0\rangle\otimes|0\rangle+b\,|1\rangle\otimes|1\rangle \tag{1.10}$$

which we call the *Bell state*, where the off-diagonal terms above vanish. Then the density operator is diagonal, with entries $|a|^2$ and $|b|^2$. The result of our expectation value is exactly what we would expect to get if we specified that system A was in state $|0\rangle$ with probability $p_0=|a|^2$ and state $|1\rangle$ with probability $p_1=|b|^2$. This is quite different from system A being in a superposition of states $|0\rangle$ and $|1\rangle$, as we illustrate with a small example.

Suppose we prepare a system in the Bell state from above with $|a|^2=|b|^2=\frac{1}{2}$, so that the density operator ρ_A is given by

$$\rho_A=\frac{1}{2}\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}=\frac{1}{2}I,$$

and it looks like an ensemble over the two equally probable states. We distinguish this from the single system in the superposition of states,

$$\psi_A=\frac{1}{\sqrt{2}}\left(|0\rangle+|1\rangle\right).$$

Then if we measure the probability for the state ψ_A from the ensemble state, we get

$$\begin{aligned}\langle P_\psi\rangle &= \mathrm{Tr}\{|\psi_A\rangle\langle\psi_A|\,\rho_A\} &(1.11)\\ &= \frac{1}{4}\,\mathrm{Tr}\{(|0\rangle\langle 0|+|0\rangle\langle 1|+|1\rangle\langle 0|+|1\rangle\langle 1|)\,(|0\rangle\langle 0|+|1\rangle\langle 1|)\} &(1.12)\\ &= \frac{1}{4}\,(1+1) &(1.13)\\ &= \frac{1}{2}, &(1.14)\end{aligned}$$

whereas if we measured the probability of the original state ψ_A to be in state ψ_A, we would of course get unity, since it is certainly in that state. In fact, we may take any unitary transformation U of ψ_A and get the same result

$$\langle P_{U\psi} \rangle = \mathrm{Tr}\{|U\psi_A\rangle\langle\psi_A| U^\dagger \rho_A\} \tag{1.15}$$

$$= \mathrm{Tr}\left\{|\psi_A\rangle\langle\psi_A| U^\dagger \frac{1}{2} I U\right\} \tag{1.16}$$

$$= \mathrm{Tr}\{|\psi_A\rangle\langle\psi_A| \rho_A\}. \tag{1.17}$$

We can now define clearly a quantum pure state, which is a single ray in the Hilbert space, or something with a density operator which has a single term (diagonal element) in the eigenbasis. We can also call a state with multiple terms in the diagonalized density matrix an *incoherent superposition*, as opposed to a *coherent superposition* which is the normal pure state we have seen before (Peres 2006).

References

Shannon, Claude Elwood (July 1948). "A Mathematical Theory of Communication". In: *Bell System Technical Journal* 27.3, pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x. URL: https://web.archive.org/web/19980715013250/http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf.

Hardy, Lucien (2001). "Quantum theory from five reasonable axioms". In: eprint: quant-ph/0101012.

Schack, Rüdiger (2003). "Quantum theory from four of Hardy's axioms". In: *Foundations of Physics* 33.10, pp. 1461–1468.

Peres, Asher (2006). *Quantum theory: concepts and methods*. Vol. 57. Springer Science & Business Media.

Chapter 2

Linear Spaces

2.1 Definition

Numerical linear algebra is one of the most developed parts of numerical analysis. It is also the solid foundation of numerical computing. The basic outline of linear algebra has been clear since at least Grassman's 1862 treatment (Fearnley-Sander 1979). A vector space V over a field F is defined by the axioms in Table 2.1, and in everything we do F will be either the real or complex numbers. In addition, linear algebra studies mappings between vector spaces that preserve the vector-space structure. Given two vector spaces V and W, a linear operator is a map $A : V \to W$ that is compatible with vector addition and scalar multiplication,

$$A(\mathbf{u} + \mathbf{v}) = A\mathbf{u} + A\mathbf{v}, \quad A(a\mathbf{v}) = aA\mathbf{v} \qquad \forall \mathbf{u}, \mathbf{v} \in V, a \in F. \tag{2.1}$$

This should have been covered in detail in your linear algebra courses.

There are two principal jobs in scientific computing: design of the interface in order to control complexity, and efficiency of the implementation. In this unit we will try to indicate why the current interface has become the standard, and what pieces of it are likely to continue going forward. In a later unit, we will analyze the runtime performance of various implementations. However, none of this can be accomplished without the ability to run a linear algebra code.

There are many well-known packages which support numerical linear algebra, including BLAS/LAPACK (Lawson et al. 1979; Anderson et al. 1990), Hypre (Falgout 2017; Falgout n.d.), Trilinos (Heroux and Willenbring 2003; Heroux et al. n.d.), DUNE (Bastian et al. 2015), Eigen (Jacob and Guennebaud 2015), and Elemental (Poulson et al. 2013; Poulson 2015). We will use the PETSc libraries (Balay, Abhyankar, et al. 2020; Balay, Abhyankar, et al. 2019; Balay, Gropp, et al. 1997) for a number of reasons. PETSc supports scalable, distributed sparse linear algebra, which will be our focus since we will be concerned with larger problems that cannot be contained in a single machine memory and mainly with PDE or graph problems which have a sparse structure.

Axiom	Signification
(A1) Associativity of addition	$\mathbf{u} + (\mathbf{v} + \mathbf{w}) = (\mathbf{u} + \mathbf{v}) + \mathbf{w}$
(A2) Commutativity of addition	$\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$
(A3) Vector identity element	$\exists \mathbf{0} \in V \mid \mathbf{v} + \mathbf{0} = \mathbf{v} \quad \forall \mathbf{v} \in V$
(A4) Vector inverse element	$\forall \mathbf{v} \in V, \exists - \mathbf{v} \in V \mid \mathbf{v} + (-\mathbf{v}) = \mathbf{0}$
(A5) Distributivity for vector addition	$a(\mathbf{u} + \mathbf{v}) = a\mathbf{u} + a\mathbf{v}$
(A6) Distributivity for field addition	$(a + b)\mathbf{v} = a\mathbf{v} + b\mathbf{v}$
(A7) Scalar and field multiplication	$a(b\mathbf{v}) = (ab)\mathbf{v}$
(A8) Scalar identity element	$1\mathbf{v} = \mathbf{v}$

Table 2.1: The definition of a vector space (Wikipedia 2015)

For dense linear algebra problems, we will use Elemental. PETSc is designed as a hierarchical set of library interfaces, and uses C to enhance both portability and language interoperability. A discussion of tradeoffs involved in language choice can be found in (Knepley 2012).

Example: Cartesian vectors We can take arrays of real or complex numbers, indicating extent in the coordinate directions, as our vectors. Addition then means adding the numbers pairwise for each coordinates, and scalar multiplication just means multiplying each entry by the same scalar. We call this space $\mathbb{R}^n$ or $\mathbb{C}^n$ if there are n coordinates.

Example: Cartesian matrices We take matrices of m rows and n columns with real or complex entries as our vectors. Addition adds the matrices entrywise, and scalar multiplication multiplies each entry. This just reproduces $\mathbb{R}^{mn}$ or $\mathbb{C}^{mn}$, and corresponds to unrolling a matrix into a vector of length mn.

Example: Polynomials Let us consider polynomials of degree k with real or complex coefficients. We will take as our vectors, each representing a particular polynomial, the arrays of $k+1$ scalar coefficients. Pointwise addition of polynomial functions then corresponds exactly to addition of the coefficients,

$$\begin{aligned} p_0(x) + p_1(x) &= \left(a_{00} + a_{01}x + \ldots + a_{0k}x^k\right) + \left(a_{10} + a_{11}x + \ldots a_{1k} + x^k\right) \\ &= (a_{00} + a_{10}) + (a_{01} + a_{11})\, x + \ldots + (a_{0k} + \ldots a_{1k})\, x^k. \end{aligned}$$

Multiplication by a scalar just multiplies each coefficient by the same number. Thus we have exactly $\mathbb{R}^k$ and $\mathbb{C}^k$. Notice that we can replace the coefficients by any field, for instance the p-adic numbers.

2.1.1 Proofs

In order to make ourselves familiar with the axioms, we will go over a few simple proofs. First we will demonstrate a *cancellation* lemma (C), namely that if two

different vectors added to the same vector produce the same result, these vectors must be equal. First we assume the hypothesis (H) that $\mathbf{u}+\mathbf{v}=\mathbf{u}+\mathbf{w}$. Then

$$\begin{aligned}
-\mathbf{u}+(\mathbf{u}+\mathbf{v}) &= -\mathbf{u}+(\mathbf{u}+\mathbf{v}) && \\
-\mathbf{u}+(\mathbf{u}+\mathbf{v}) &= -\mathbf{u}+(\mathbf{u}+\mathbf{w}) && (H)\\
(-\mathbf{u}+\mathbf{u})+\mathbf{v} &= (-\mathbf{u}+\mathbf{u})+\mathbf{w} && (A1)\\
(\mathbf{u}+-\mathbf{u})+\mathbf{v} &= (\mathbf{u}+-\mathbf{u})+\mathbf{w} && (A2)\\
\mathbf{0}+\mathbf{v} &= \mathbf{0}+\mathbf{w} && (A4)\\
\mathbf{v} &= \mathbf{w} && (A3)
\end{aligned}$$

This result can be used to prove many things. For example, the zero vector is unique because if we had another vector $\mathbf{0}'$ such that

$$\mathbf{v}+\mathbf{0}=\mathbf{v}=\mathbf{v}+\mathbf{0}',$$

then we can use the cancellation lemma to conclude

$$\mathbf{0}=\mathbf{0}'.$$

This same proof can allow us to conclude that the inverse is unique by considering a vector $-\mathbf{v}'$,

$$\mathbf{v}+-\mathbf{v}=\mathbf{0}=\mathbf{v}+-\mathbf{v}',$$

and again using the cancellation lemma we have

$$-\mathbf{v}=-\mathbf{v}'.$$

We can also prove (Z) that multiplication by scalar 0 gives us the zero vector $\mathbf{0}$,

$$\begin{aligned}
\mathbf{0}+0\mathbf{v} &= 0\mathbf{v} && (A3)\\
\mathbf{0}+0\mathbf{v} &= (0+0)\mathbf{v} && \\
\mathbf{0}+0\mathbf{v} &= 0\mathbf{v}+0\mathbf{v} && (A5)\\
\mathbf{0} &= 0\mathbf{v} && (C)
\end{aligned}$$

and that multiplication by negative one gives the inverse,

$$\begin{aligned}
\mathbf{v}+-1\mathbf{v} &= 1\mathbf{v}+-1\mathbf{v} && (A8)\\
\mathbf{v}+-1\mathbf{v} &= (1-1)\mathbf{v} && (A5)\\
\mathbf{v}+-1\mathbf{v} &= 0\mathbf{v} && \\
\mathbf{v}+-1\mathbf{v} &= \mathbf{0} && (Z)\\
\mathbf{v}+-1\mathbf{v} &= \mathbf{v}+-\mathbf{v} && (A4)\\
-1\mathbf{v} &= -\mathbf{v} && (C)
\end{aligned}$$

2.1.2 Useful notation

We will indicate the ith entry of a vector $\mathbf{v}$ with v_i, so that

$$\mathbf{v} = \sum_i v_i \hat{\mathbf{e}}_i \tag{2.2}$$

where $\hat{\mathbf{e}}_i$ is the ith basis vector. The *Kronecker delta* function, written δ_{ij} is zero if the two indices are different and unity if they are the same. For example, the entries of the identity matrix I can be expressed as

$$I_{ij} = \delta_{ij}. \tag{2.3}$$

The Kronecker delta is a very useful device for manipulating indices and representing matrices whose only entries are 0 and 1, such as permutation matrices. Another very useful notation is the *Einstein summation notation*. This declares that repeated indices should be summed over. For example,

$$\begin{aligned} I_{ii} &= \sum_i \delta_{ii} \\ &= \sum_i 1 \\ &= \mathrm{Tr}\{I\} \end{aligned}$$

and in general

$$A_{ii} = \mathrm{Tr}\{A\}.$$

A more complex example would be matrix multiplication, so that $C = AB$ could be expressed as

$$C_{ij} = A_{ik} B_{kj}$$

and also

$$A_{ik} B_{ki} = \mathrm{Tr}\{AB\}.$$

2.2 Inner Products, Orthogonality, and Dual Spaces

We can impose some additional structure on our vector space, namely that we can compare angles between vectors. We will define the *inner product* of two vectors as

$$\mathbf{w} \cdot \mathbf{v} = \sum_i w_i v_i. \tag{2.4}$$

However, we will not use this notation very often, since there is a complication for complex vector spaces. Instead, we will connect the idea of the inner product with that of a dual space, arriving at a more compact and useful notation.

We will define the *dual space* $V^\dagger$ as the space of linear functionals on our vector space V. This means the space of linear mappings from V into the field of scalars for our vector space, usually $\mathbb{R}$ or $\mathbb{C}$. According to the famous Riesz Representation Theorem, the space $V^\dagger$ is isomorphic to V, and we can represent the action of any functional $\psi \in V^\dagger$ on a vector $\mathbf{v} \in V$ by the inner product of some vector $\mathbf{w}$ with $\mathbf{v}$, so that

$$\psi(\mathbf{v}) = \bar{\mathbf{w}} \cdot \mathbf{v} = \sum_i \bar{w}_i v_i. \tag{2.5}$$

Now we define the *Hermitian conjugate* of a vector $w^\dagger$ so that

$$\mathbf{w}^\dagger \mathbf{v} = \sum_i \bar{w}_i v_i. \tag{2.6}$$

Thus, the Hermitian conjugate finds the functional represented by that vector. Sometimes people explain this as having "row" and "column" vectors, but this a cumbersome and fragile way to explain things.

If we take the inner product of a vector with itself, we get the square of its length

$$\|\mathbf{v}\| = \sqrt{\mathbf{v}^\dagger \mathbf{v}} \tag{2.7}$$

which is also the 2-norm of the vector, discussed later on. With this, we can define the angle α between two vectors as

$$\cos\alpha = \frac{\mathbf{w}^\dagger \mathbf{v}}{\|\mathbf{v}\|\|\mathbf{w}\|}. \tag{2.8}$$

Clearly, vectors whose inner product is zero correspond to $\alpha = \pi/2$ or a right angle. We call these vectors *orthogonal*. The most important use of orthogonality is to form bases for the span of a set of linearly independent vectors.

2.3 Bases

A *linear combination* of vectors is the sum

$$\alpha_0 \mathbf{v}_0 + \alpha_1 \mathbf{v}_1 + \cdots + \alpha_n \mathbf{v}_n = \sum_{i=0}^{n} \alpha_i \mathbf{v}_i \tag{2.9}$$

where each α_i is a scalar from some field. The only fields we will use in this class are the real numbers $\mathbb{R}$ and the complex numbers $\mathbb{C}$. The *span* of a set of vectors $\{\mathbf{v}_i\}$ is the subspace of vectors which can be constructed as linear combinations of $\{\mathbf{v}_i\}$. In the quantum mechanics literature, a linear combination of states is called a *superposition*. A *linearly independent* set is a set of vectors where the only linear combination that vanishes, namely

$$\lambda_0 \mathbf{v}_0 + \lambda_1 \mathbf{v}_1 + \cdots \lambda_n \mathbf{v}_n = 0, \tag{2.10}$$

requires that

$$\lambda_0 = \lambda_1 = \cdots = \lambda_n = 0. \tag{2.11}$$

Thus no combination of linearly independent vectors can add up to zero.

A basis $\{\hat{\mathbf{e}}_i\}$ for a vector space V is a set of linearly independent vectors whose span is the entire space, such that every vector $\mathbf{v} \in V$ can be written as a finite linear combination of $\hat{\mathbf{e}}_i$ in a unique way. I am ignoring subtleties connected with infinite dimensional vector spaces, as I will for this entire course. A simple procedure to construct a basis is to start with one vector, which is a trivial linearly independent set. If that vector spans the space, we are done. If not, add a vector which is not in its span, and repeat. If V is finite dimensional, then this process is guaranteed to terminate in d steps, where d is the dimension of the space.

Given that every vector $\mathbf{v} \in V$ can be expressed as a linear combination of basis vectors

$$\mathbf{v} = \sum_i v_i \hat{\mathbf{e}}_i,$$

how do we find the scalars v_i? We can take the inner product of the equation above with some basis vector $\hat{\mathbf{e}}_k$, so that

$$\hat{\mathbf{e}}_k^\dagger \mathbf{v} = \sum_i v_i \hat{\mathbf{e}}_k^\dagger \hat{\mathbf{e}}_i, \tag{2.12}$$

$$\bar{v}_k = \sum_i v_i e_{ki}, \tag{2.13}$$

where we defined $\bar{v}_k = \hat{\mathbf{e}}_k^\dagger \mathbf{v}$ and $e_{ki} = \hat{\mathbf{e}}_k^\dagger \hat{\mathbf{e}}_i$, both of which may be calculated if we know the basis and vector. This may be recast in linear algebraic notation as a matrix equation

$$E\mathbf{v} = \bar{\mathbf{v}} \tag{2.14}$$

$$\mathbf{v} = E^{-1}\bar{\mathbf{v}} \tag{2.15}$$

where $\mathbf{v}$ is the vector of coefficients v_i, $\bar{\mathbf{v}}$ is the vector of coefficients $\bar{v}_i$, and E is the matrix of coefficients e_{ij}. Now if we have an orthonormal basis, which we will assume from here on, then

$$e_{ij} = \hat{\mathbf{e}}_i^\dagger \hat{\mathbf{e}}_j = \delta_{ij}, \tag{2.16}$$

which means that

$$v_i = \bar{v}_i = \hat{\mathbf{e}}_i^\dagger \mathbf{v}. \tag{2.17}$$

Suppose instead that we have two different bases, $\{\hat{\mathbf{e}}_i\}$ and $\{\hat{\mathbf{f}}_i\}$. How would we get the components v_i^f of some vector $\mathbf{v}$ in the f-basis if we already know the compnents v_i^e in the e-basis? This is a practical problem, since we often measure

in some basis but do calculations in another. We can derive an expression for this by expanding basis vectors of the first set in terms of basis vector in the second

$$\mathbf{v} = \sum_i v_i^e \hat{\mathbf{e}}_i \tag{2.18}$$

$$\sum_i v_i^f \mathbf{f}_i = \sum_i v_i^e \hat{\mathbf{e}}_i \tag{2.19}$$

$$\sum_i v_i^f \mathbf{f}_i = \sum_i v_i^e \sum_j V_{ji} \mathbf{f}_j \tag{2.20}$$

where $V_{ji} = \mathbf{f}_j \cdot \hat{\mathbf{e}}_i$ is the j-coefficient of the basis vector $\hat{\mathbf{e}}_i$ in the f-basis. Now we can apply $\mathbf{f}_k^\dagger$, which is the same as the dot product with $\mathbf{f}_k$,

$$\sum_i v_i^f \mathbf{f}_k^\dagger \mathbf{f}_i = \sum_i v_i^e \sum_j V_{ji} \mathbf{f}_k^\dagger \mathbf{f}_j, \tag{2.21}$$

$$\sum_i v_i^f \delta_{ki} = \sum_i v_i^e \sum_j V_{ji} \delta_{kj}, \tag{2.22}$$

$$v_k^f = \sum_i v_i^e V_{ki}, \tag{2.23}$$

$$\mathbf{v}^f = V \mathbf{v}^e. \tag{2.24}$$

Thus the coefficients in the f-basis can be obtained from the coefficients in the e-basis by applying the matrix V with coefficients

$$V_{ij} = \mathbf{f}_i \cdot \hat{\mathbf{e}}_j \tag{2.25}$$

which we will call the *Vandermonde matrix*, although Vandermonde was original talking about a very specific change of basis (see Problem 2).

2.3.1 Orthogonalization

In order to introduce the Gram-Schmidt orthogonalization method, we follow an excellent presentation by Per-Olof Persson. The purpose of orthogonalization is to take a set of linearly independent vectors and produce an orthogonal set with the same span. In fact, we could start with a set of linearly dependent vectors, but we would produce an orthogonal set which was smaller than the original.

The idea for the Gram-Schmidt process is quite simple. Take each vector in turn, cut off the pieces in the direction of any prior vector, and normalize the result. Since the new vector has no part in the direction of any prior vector, it is orthogonal to them. We can see that the correct piece to remove is the projection of the candidate vector v_j onto the basis vector q_i,

$$\begin{aligned} v_j' &= v_j - q_i q_i^\dagger v_j \\ &= \left(I - q_i q_i^\dagger\right) v_j \end{aligned}$$

input : A set of n vectors $\{a_i\}$
output: A set of orthonormal vectors $\{q_i\}$ with the same span

for *j = 1 to n* **do**
  $v_j = a_j$;
  for *i = 1 to j-1* **do**
    $r_{ij} = q_i^\dagger a_j$ (CGS);
    $r_{ij} = q_i^\dagger v_j$ (MGS);
    $v_j = v_j - r_{ij} q_i$;
  end
  $r_{jj} = \|v_j\|_2$;
  $q_j = v_j / r_{jj}$;
end

Algorithm 1: Gram-Schmidt Orthogonalization

which is the same as projecting v_j into the space orthogonal to q_i using the complementary projector. We produce the pseudocode for this in Algorithm 1.

It should be clear from the naming scheme that this produces exactly the QR factorization of the a matrix A with columns equal to our input set $\{a_i\}$, $A = QR$. Note that we have included two variants of the algorithm, the Classical Gram-Schmidt (CGS) and the Modified Gram-Schmidt (MGS). The classical version calculates all projections of the original vector onto the basis vectors, and then subtracts these pieces at once. The modified variant subtracts the projection onto basis vector q_i from the candidate vector before calculating the projection on q_{i+1}. This may seem like a small detail, however the example below will show that CGS can suffer from catastrophic loss of accuracy when applying the projector.

Consider the matrix A to be orthogonalized,

$$A = \begin{pmatrix} 1 & 1 & 1 \\ \epsilon & 0 & 0 \\ 0 & \epsilon & 0 \\ 0 & 0 & \epsilon \end{pmatrix}.$$

where we will assume that ϵ is so small that ϵ^2 can be neglected since it will not be resolvable by our machine arithmetic. Our first column a_1 has norm $1 + \epsilon^2 \approx 1$, so we have $q_1 = a_1$ at the first iteration. In the second iteration,

$$r_{12} = q_1^\dagger a_2 = 1$$

in both variants, so that

$$v_2 = v_2 - r_{12} q_1 = \begin{pmatrix} 0 \\ -\epsilon \\ \epsilon \\ 0 \end{pmatrix}.$$

Thus we have $r_{22} = \epsilon\sqrt{2}$ giving

$$q_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ -1 \\ 1 \\ 0 \end{pmatrix}.$$

The third iteration is where things diverge between variants. We begin with

$$r_{13} = q_1^\dagger a_3 = 1,$$

so that we update the v_3 vector as

$$v_3 = a_3 - r_{13} q_1 = \begin{pmatrix} 0 \\ -\epsilon \\ 0 \\ \epsilon \end{pmatrix}.$$

Now the two variants give a different answer for the coefficient r_{23}

$$r_{23} = \begin{cases} q_2^\dagger a_3 = 0 & \text{CGS} \\ q_2^\dagger v_3 = \epsilon/\sqrt{2} & \text{MGS} \end{cases}$$

which leads to two different updates

$$v_3 = v_3 - 0 \cdot q_2 = \begin{pmatrix} 0 \\ -\epsilon \\ 0 \\ \epsilon \end{pmatrix} \text{ vs } v_3 = v_3 - \epsilon/\sqrt{2} \cdot q_2 = \frac{1}{2} \begin{pmatrix} 0 \\ -\epsilon \\ -\epsilon \\ 2\epsilon \end{pmatrix}$$

leading to different normalizations $r_{33} = \epsilon\sqrt{2}$ vs $\epsilon\sqrt{\frac{3}{2}}$, and finally

$$q_3 = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ -1 \\ 0 \\ 1 \end{pmatrix} \text{ vs } q_3 = \frac{1}{\sqrt{6}} \begin{pmatrix} 0 \\ -1 \\ -1 \\ 2 \end{pmatrix}.$$

This difference has devastating consequences for orthogonality, as we see that

$$q_2^\dagger q_3 = \frac{1}{2} \text{ vs } q_2^\dagger q_3 = 0.$$

To get a better idea how the code actually performs, we can look at the PETSc implementation of both variants. In PETSc itself, orthogonalization is only used in the context of Krylov solvers like GMRES. the set of vectors are stored in the `VEC_VV` array and the coefficients r_{ij} are stored in `hes` which indicates the Hesseneberg matrix constructed by GMRES. We just look at the inner loop, acting on vector a_{it+1}. The modified variant is much easier to recognize, looking almost exactly like our pseudocode.

```
for (j = 0; j <= it; ++j) {
 /* (vv(it+1), vv(j)) */
 ierr = VecDot(VEC_VV(it+1), VEC_VV(j), hh);CHKERRQ(ierr);
 *hes++ = *hh;
 /* vv(it+1) <- vv(it+1) - hh[it+1][j] vv(j) */
 ierr = VecAXPY(VEC_VV(it+1), -(*hh++), VEC_VV(j));CHKERRQ(ierr);
}
```

On the other hand, the classical variant takes advantage of the fact that all the dot products from the projectors can be executed simultaneously, as can all the subtractions. The `VecMAXPY` call can vectorize of the input vectors, as well as only load the candidate vector once, saving memory bandwidth. The `VecMDot` can use a single reduction for all dot products, reducing the latency of the operation. These savings can be important, especially in a parallel code.

```
ierr = VecMDot(VEC_VV(it+1),it+1,&(VEC_VV(0)),lhh);CHKERRQ(ierr); /* <v,vnew> */
for (j = 0; j <= it; ++j) {
  lhh[j] = -lhh[j];
}
ierr = VecMAXPY(VEC_VV(it+1), it+1, lhh, &VEC_VV(0));CHKERRQ(ierr);
/* note lhh[j] is -<v,vnew> , hence the subtraction */
for (j = 0; j <= it; ++j) {
  hh[j] -= lhh[j]; /* hh += <v,vnew> */
  hes[j] -= lhh[j]; /* hes += <v,vnew> */
}
```

We can examine the benefit with a simple performance model. For an $m \times n$ matrix, each norm would cost $2m$ flops and m memory references, along with m flops and m references for the normalization. However, this cost is lower order. In the inner loop, we do $2m$ flops and $2m$ memory references for each dot product, and $2m$ flops and $3m$ memory references for each subtraction. We execute the inner loop $n(n-1)/2$ times, which means that in total, we have

$$(2mn(n-1) + 3m) \text{ flops} \qquad \text{and} \qquad \left(\frac{5}{2}mn(n-1) + 2m\right) b \text{ bytes}$$

Thus the *arthmetic intensity*, the ratio of flops to bytes for our algorithm, is given by

$$\frac{4mn(n-1) + 6m}{(5mn(n-1) + 4m)b} \approx \frac{4}{5b}$$

which in double precision gives a miserable 0.1 flops/byte. This means that our orthogonalization computation will be bandwidth bound on any modern architecture. However, if we take advantage of the multiple dot product and vector subtraction functions in PETSc, we can reduce the bandwidth cost since

we only load and store the candidate vector once. This means that the required bandwidth is now

$$\left(\frac{2}{2}mn(n-1)+5m\right)b \text{ bytes}$$

so that the final arithmetic intensity is more than twice as large,

$$\frac{4mn(n-1)+6m}{(2mn(n-1)+10m)b} \approx \frac{2}{b}.$$

In addition, this algorithm has lower latency since we use a single reduction for all the dot products.

2.4 Linear Operators

2.4.1 Expansion in a Basis

Matrix multiplication is simply the application of a linear operator A between two vector spaces, to an input vector $\mathbf{x}$, generating an output vector $\mathbf{y}$,

$$A\mathbf{x} = \mathbf{y}. \tag{2.26}$$

This operation is required to be linear, namely

$$A(\alpha\mathbf{x} + \beta\mathbf{z}) = \alpha(A\mathbf{x}) + \beta(A\mathbf{z}). \tag{2.27}$$

If we expand the vectors $\mathbf{x}$ and $\mathbf{y}$ in some basis $\{\hat{\mathbf{e}}\}$, noting that a basis is guaranteed to exist for any Hilbert space, we have

$$x = \sum_j x_j\hat{\mathbf{e}}_j \quad \text{and} \quad y = \sum_j y_j\hat{\mathbf{e}}_j, \tag{2.28}$$

and plugging into Eq. (2.26) gives

$$\begin{aligned} A\sum_j x_j\hat{\mathbf{e}}_j &= \sum_j y_j\hat{\mathbf{e}}_j, \\ \sum_j x_j(A\hat{\mathbf{e}}_j) &= \sum_j y_j\hat{\mathbf{e}}_j. \end{aligned} \tag{2.29}$$

Now we suppose that our basis is orthonormal, meaning that

$$\hat{\mathbf{e}}_i \cdot \hat{\mathbf{e}}_j = \delta_{ij}. \tag{2.30}$$

We can take the inner product of Eq. 2.29 with $\hat{\mathbf{e}}_i$,

$$\begin{aligned}
\hat{\mathbf{e}}_i \cdot \sum_j x_j (A\hat{\mathbf{e}}_j) &= \hat{\mathbf{e}}_i \cdot \sum_j y_j \hat{\mathbf{e}}_j, \\
\sum_j x_j \hat{\mathbf{e}}_i \cdot (A\hat{\mathbf{e}}_j) &= \sum_j y_j \hat{\mathbf{e}}_i \cdot \hat{\mathbf{e}}_j, \\
\sum_j x_j \hat{\mathbf{e}}_i \cdot (A\hat{\mathbf{e}}_j) &= \sum_j y_j \delta_{ij}, \\
\sum_j a_{ij} x_j &= y_i.
\end{aligned} \tag{2.31}$$

where used the linearity of the inner product in line 2, the orthogonality of basis vectors from Eq. 2.30, and we defined the matrix elements

$$a_{ij} = \hat{\mathbf{e}}_i \cdot (A\hat{\mathbf{e}}_j). \tag{2.32}$$

We see that Eq. 2.31 is exactly our rule for matrix multiplication, but we have derived it from the properties of abstract linear operators and bases. This means that we can use our insights in domains others than the Euclidean space $\mathbb{R}^n$, such as vector spaces of functions.

We will define the Hermitian conjugate, or *adjoint*, $A^\dagger$ of the operator A such that

$$\left(w^\dagger A v\right)^\dagger = v^\dagger A^\dagger w. \tag{2.33}$$

Since we have defined the matrix element $a_{ij} = \hat{\mathbf{e}}_i^\dagger A \hat{\mathbf{e}}_j$, it means that the matrix element $a_{ij}^\dagger$ for the Hermitian conjugate $A^\dagger$ should be

$$a_{ij}^\dagger = \hat{\mathbf{e}}_i^\dagger A^\dagger \hat{\mathbf{e}}_j = \left(\hat{\mathbf{e}}_j^\dagger A \hat{\mathbf{e}}_i\right)^\dagger = a_{ji}^\dagger = \bar{a}_{ji}. \tag{2.34}$$

Thus we get the Hermitian conjugate of a matrix by interchanging rows and columns and taking the complex conjugate. If the matrix is real, then we call this the *transpose*. Suppose that we want the Hermitian conjugate of the product of two matrices AB, then

$$(AB)_{ij}^\dagger = \left(\sum_k a_{ik} b_{kj}\right)^\dagger \tag{2.35}$$

$$= \sum_k \bar{a}_{jk} \bar{b}_{ki} \tag{2.36}$$

$$= \sum_k B_{ik}^\dagger A_{kj}^\dagger \tag{2.37}$$

$$= \left(B^\dagger A^\dagger\right)_{ij} \tag{2.38}$$

so that $AB^\dagger = B^\dagger A^\dagger$. A similar thing can be proved for inverses, in a simpler way,

$$I = AB\,(AB)^{-1} \tag{2.39}$$
$$= ABB^{-1}A^{-1} \tag{2.40}$$
$$= AA^{-1} \tag{2.41}$$
$$= I. \tag{2.42}$$

As practice working with matrices, consider the multiplication of two upper triangular matrices R and R', so that

$$R'' = RR'$$
$$R''_{ij} = \sum_k R_{ik} R'_{kj}$$

Suppose that $i > j$. If $k > j$, $R'_{kj} = 0$, so if R''_{ij} is nonzero, then $k \leq j$. However for $R_{ik} \neq 0$, we need $i \leq k$, which together with $k \leq j$ implies $i \leq j$. This contradicts our assumption, so that $R''_{ij} = 0$ for $i > j$ and R'' is upper triangular. Thus the product of upper triangular matrices is upper triangular. By the same argument we can also show that the product of lower triangular matrices is lower triangular.

2.4.2 Unitary operators

A *unitary* operator U is defined by

$$UU^\dagger = U^\dagger U = I. \tag{2.43}$$

Notice that this implies that the columns of U are orthonormal, since

$$\left(U^\dagger U\right)_{ij} = \sum_k u^\dagger_{ik} u_{kj} \tag{2.44}$$
$$= \sum_k \bar{u}_{ki} u_{kj} \tag{2.45}$$
$$= \bar{u}_i \cdot u_j \tag{2.46}$$
$$= \delta_{ij} \tag{2.47}$$

where u_i is the ith column of U. A similar calculation with the transpose identity shows that the rows are also orthogonal. A unitary transformation is an *isometry*, meaning a transformation which preserves the metric on a space or the norm of every vector. More precisely, unitary operators are L_2 isometries because they preserve the 2-norm of vectors,

$$\|Ux\|_2^2 = (Ux)^\dagger (Ux) \tag{2.48}$$
$$= x^\dagger U^\dagger U x \tag{2.49}$$
$$= x^\dagger x \tag{2.50}$$
$$= \|x\|_2^2. \tag{2.51}$$

A very common type of unitary operator is a permutation matrix, which a single one in each row. The row represents the new index and the column the old index. Since applying the permutation followed by the inverse permutation gives the identity, it is unitary. We can show this by using the Kronecker delta to express the elements of a permutation matrix P,

$$P_{ij} = \delta_{i\sigma(j)} \tag{2.52}$$

where $\sigma(k)$ is the permutation function, giving the index for element k after permutation. We can see this by acting on the basis vector $\hat{\mathbf{e}}_k$ with P,

$$(P\hat{\mathbf{e}}_k)_i = \sum_j P_{ij}(\hat{\mathbf{e}}_k)_j \tag{2.53}$$

$$= \sum_j \delta_{i\sigma(j)}\delta_{kj} \tag{2.54}$$

$$= \delta_{i\sigma(k)} \tag{2.55}$$

so that the output is $\hat{\mathbf{e}}_{\sigma(k)}$. Now we can look at the matrix product

$$\left(P^\dagger P\right)_{ij} = \sum_k P^\dagger_{ik} P_{kj} \tag{2.56}$$

$$= \sum_k P_{ki} P_{kj} \tag{2.57}$$

$$= \sum_k \delta_{k\sigma(i)}\delta_{k\sigma(j)} \tag{2.58}$$

$$= \delta_{\sigma(i)\sigma(j)} \tag{2.59}$$

$$= \delta_{ij} \tag{2.60}$$

where the last step follows because σ is one-to-one. Thus $P^\dagger P = I$ and P is unitary.

2.4.3 Block Matrices

As a consequence of linearity, we can simplify the presentation of matrices with block structure. Consider the 4×4 matrix and vector

$$A = \left(\begin{array}{cc|cc} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ \hline a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{array}\right) \qquad x = \left(\begin{array}{c} x_0 \\ x_1 \\ \hline x_2 \\ x_3 \end{array}\right).$$

The expression for Ax is given by

$$(Ax)_i = \sum_j a_{ij} x_j,$$

but we can express the loop over $j \in [0,4)$ as two loops by splitting the index into subindices $j = k * 2 + l$ for $k, l \in [0,2)$,

$$\begin{aligned}(Ax)_i &= \sum_k \sum_l a_{i,k*2+l} x_{k*2+l}, \\ &= \sum_k \sum_l a_{i,(kl)} x_{(kl)},\end{aligned}$$

where (kl) is multindex, k indicating which block and l the index within that block. Now suppose we also index the output vector using our multindex (ij), so that

$$\begin{aligned}(Ax)_{(ij)} &= \sum_k \sum_l a_{(ij),(kl)} x_{(kl)}, \\ (Ax)_i &= \sum_k a_{i,k} x_k,\end{aligned}$$

where

$$a_{i,k} x_k = \begin{pmatrix} a_{(i0),(k0)} & a_{(i0),(k1)} \\ a_{(i1),(k0)} & a_{(i1),(k1)} \end{pmatrix} \begin{pmatrix} x_{(k0)} \\ x_{(k1)} \end{pmatrix}$$

so that our usual rule for matrix-vector multiplication applies to the individual blocks, and we can write

$$A = \left(\begin{array}{c|c} a_{00} & a_{01} \\ \hline a_{10} & a_{11} \end{array}\right) \qquad x = \left(\begin{array}{c} x_0 \\ \hline x_1 \end{array}\right),$$

where each entry is a small vector or matrix, and multiplication is understood to be matrix-vector multiplication. This same procedure extends to matrix-matrix multiplication, and on to more general tensors.

2.5 Tensor Product Spaces

The *tensor product* $V \otimes W$ of two vector spaces V and W (over the same field) is itself a vector space, together with an operation of bilinear composition, denoted by $\otimes$, from ordered pairs in the Cartesian product $V \times W$ into $V \otimes W$. The tensor product is defined by the bilinearity of the product operation $\otimes$,

$$\begin{aligned} \forall \mathbf{v} \in V, \forall \mathbf{w}_0, \mathbf{w}_1 \in W \qquad & \mathbf{v} \otimes (\alpha_0 \mathbf{w}_0) + \mathbf{v} \otimes (\alpha_1 \mathbf{w}_1) = \mathbf{v} \otimes (\alpha_0 \mathbf{w}_0 + \alpha_1 \mathbf{w}_1), \\ \forall \mathbf{v}_0, \mathbf{v}_1 \in V, \forall \mathbf{w} \in W \qquad & (\alpha_0 \mathbf{v}_0) \otimes \mathbf{w} + (\alpha_1 \mathbf{v}_1) \otimes \mathbf{w} = (\alpha_0 \mathbf{v}_0 + \alpha_1 \mathbf{v}_1) \otimes \mathbf{w}. \end{aligned}$$

Given two linear operators $A : V \to X$ and $B : W \to Y$, we define the tensor product of the operators as a linear map

$$A \otimes B : V \otimes W \to X \otimes Y \tag{2.61}$$

such that

$$(A \otimes B)(\mathbf{v} \otimes \mathbf{w}) = (A\mathbf{v}) \otimes (B\mathbf{w}), \tag{2.62}$$

which also implies that

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD). \tag{2.63}$$

We can get the action of the combined operator on a combined vector by inserting bases for the two spaces $\{\hat{\mathbf{e}}_i\}$ and $\{\hat{\mathbf{f}}_i\}$,

$$(A \otimes B)(\sum_i v_i \hat{\mathbf{e}}_i \otimes \sum_j w_j \hat{\mathbf{f}}_j) = (A \sum_i v_i \hat{\mathbf{e}}_i) \otimes (B \sum_j w_j \hat{\mathbf{f}}_j), \tag{2.64}$$

$$(A \otimes B) \left(\sum_i \sum_j v_i w_j (\hat{\mathbf{e}}_i \otimes \hat{\mathbf{f}}_j) \right) = (\sum_i v_i A\hat{\mathbf{e}}_i) \otimes (\sum_j w_j B\hat{\mathbf{f}}_j). \tag{2.65}$$

Then we can get a matrix representation of the combined operator if we let the input vector be a tensor product of the basis vectors,

$$(A \otimes B) \left(\hat{\mathbf{e}}_i \otimes \hat{\mathbf{f}}_j \right) = (A\hat{\mathbf{e}}_i) \otimes (B\hat{\mathbf{f}}_j), \tag{2.66}$$

and look at the (kl) entry of the output vector by taking the dot product with that basis vector,

$$\left(\hat{\mathbf{e}}_k \otimes \hat{\mathbf{f}}_l \right)^\dagger (A \otimes B) \left(\hat{\mathbf{e}}_i \otimes \hat{\mathbf{f}}_j \right) = \left(\hat{\mathbf{e}}_k \otimes \hat{\mathbf{f}}_l \right)^\dagger \left((A\hat{\mathbf{e}}_i) \otimes (B\hat{\mathbf{f}}_j) \right), \tag{2.67}$$

$$(A \otimes B)_{(kl),(ij)} = (\hat{\mathbf{e}}_k^\dagger A \hat{\mathbf{e}}_i)(\hat{\mathbf{f}}_l^\dagger B \hat{\mathbf{f}}_j), \tag{2.68}$$

$$= a_{ki} b_{lj} \tag{2.69}$$

which is precisely the *Kronecker product* of matrices A and B, defined here. Notice that we have made a block matrix of exactly the type we saw in Section 2.4.3. For example, the Krocker product of two 2×2 matrices is given by

$$\begin{pmatrix} a_{0,0} & a_{0,1} \\ a_{1,0} & a_{1,1} \end{pmatrix} \otimes \begin{pmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{pmatrix} \tag{2.70}$$

$$= \begin{pmatrix} a_{0,0} \begin{pmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{pmatrix} & a_{0,1} \begin{pmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{pmatrix} \\ a_{1,0} \begin{pmatrix} b_{0,1} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{pmatrix} & a_{1,1} \begin{pmatrix} b_{0,0} & b_{0,1} \\ b_{1,0} & b_{1,1} \end{pmatrix} \end{pmatrix} \tag{2.71}$$

$$= \begin{pmatrix} a_{0,0}b_{0,0} & a_{0,0}b_{0,1} & a_{0,1}b_{0,0} & a_{0,1}b_{0,1} \\ a_{0,0}b_{1,0} & a_{0,0}b_{1,1} & a_{0,1}b_{1,0} & a_{0,1}b_{1,1} \\ a_{1,0}b_{0,0} & a_{1,0}b_{0,1} & a_{1,1}b_{0,0} & a_{1,1}b_{0,1} \\ a_{1,0}b_{1,0} & a_{1,0}b_{1,1} & a_{1,1}b_{1,0} & a_{1,1}b_{1,1} \end{pmatrix}. \tag{2.72}$$

In all of our quantum computing examples, we will be looking at combinations of 2-state quantum systems, so that all our tensor product operators will look

like this. Note that A is indexed with the high bit and B the low bit. If we have a tensor product of several 2×2 operators, then each one will be indexed by a given bit of the global index.

Since the action of tensor product operators can be decomposed into action on separate spaces, we can establish useful theorems about them. For example, using our definition above for adjoints, we see that

$$(A \otimes B)^\dagger = A^\dagger \otimes B^\dagger. \tag{2.73}$$

Suppose that we have the tensor product of two unitary operators. Is it also unitary? We can prove this using Eq. (2.73),

$$\begin{aligned}
(U_1 \otimes U_2)^\dagger (U_1 \otimes U_2) &= \left(U_1^\dagger \otimes U_2^\dagger\right)(U_1 \otimes U_2) && (2.74)\\
&= \left(U_1^\dagger U_1 \otimes U_2^\dagger U_2\right) && (2.75)\\
&= (I \otimes I) && (2.76)\\
&= I. && (2.77)
\end{aligned}$$

2.6 Norms

2.6.1 Vector Norms

A *norm* is a generalization of the concept of the length of a vector, and has three defining properties. The norm $\|\mathbf{v}\|$ or a vector $\mathbf{v}$ must always be non-negative

$$\|\mathbf{v}\| \geq 0, \tag{2.78}$$

and further it is only zero if $\mathbf{v} = \mathbf{0}$. This coincides with our idea that distance is always positive, and that adding the null vector to another should not change its length. Second, the norm is scaled when we multiply the vector by a scalar,

$$\|\alpha \mathbf{v}\| = |\alpha| \|\mathbf{v}\|. \tag{2.79}$$

We would expect that two copies of a vector would have twice the length, and also that the additive inverse, $-\mathbf{v}$, would have the same length but oppositely oriented. Lastly, our norm must satisfy the *triangle inequality*,

$$\|\mathbf{u} + \mathbf{v}\| \leq \|\mathbf{u}\| + \|\mathbf{v}\|, \tag{2.80}$$

which follows from the idea that the length of two sides of a triangle are always longer than the third.

We can verify these properties for our usually idea of Euclidean distance,

$$\|\mathbf{v}\|_2 = \left(\sum_i |v_i|^2\right)^{1/2} = \sqrt{\mathbf{v}^\dagger \mathbf{v}}, \tag{2.81}$$

which we will refer to as the 2-norm. Clearly the sum of squares must be non-negative, and if we scale the vector

$$\begin{aligned}\|\alpha\mathbf{v}\|_2 &= \left(\sum_i |\alpha v_i|^2\right)^{1/2} \\ &= \left(|\alpha|^2 \sum_i |v_i|^2\right)^{1/2} \\ &= |\alpha| \left(\sum_i |v_i|^2\right)^{1/2} \\ &= |\alpha| \|\mathbf{v}\|_2\end{aligned}$$

To prove the triangle inequality, we first another lemma known as the *Cauchy-Schwarz Inequality*. In order to prove this,

$$\left|\mathbf{u}^\dagger \mathbf{v}\right| \leq \|\mathbf{u}\|_2 \|\mathbf{v}\|_2, \tag{2.82}$$

we start by inserting a linear combination of vectors into Eq. (2.78),

$$0 \leq \|\mathbf{u} - \lambda \mathbf{v}\|_2^2 \tag{2.83}$$

$$= \mathbf{u}^\dagger \mathbf{u} - \lambda \mathbf{u}^\dagger \mathbf{v} - \bar{\lambda} \mathbf{v}^\dagger \mathbf{u} + |\lambda|^2 \mathbf{v}^\dagger \mathbf{v}. \tag{2.84}$$

If $\mathbf{v} = \mathbf{0}$ in Eq. (2.82), it is trivially true with equality. Thus, we assume that $\mathbf{v}$ is nonzero, and define

$$\lambda = \frac{\mathbf{v}^\dagger \mathbf{u}}{\|\mathbf{v}\|_2^2}$$

so that

$$0 \leq \mathbf{u}^\dagger \mathbf{u} - \frac{\mathbf{v}^\dagger \mathbf{u}}{\|\mathbf{v}\|_2^2} \mathbf{u}^\dagger \mathbf{v} - \frac{\mathbf{u}^\dagger \mathbf{v}}{\|\mathbf{v}\|_2^2} \mathbf{v}^\dagger \mathbf{u} + \frac{\left|\mathbf{v}^\dagger \mathbf{u}\right|^2}{\|\mathbf{v}\|_2^4} \mathbf{v}^\dagger \mathbf{v} \tag{2.85}$$

$$= \mathbf{u}^\dagger \mathbf{u} - \frac{\left|\mathbf{v}^\dagger \mathbf{u}\right|^2}{\|\mathbf{v}\|_2^2} - \frac{\left|\mathbf{v}^\dagger \mathbf{u}\right|^2}{\|\mathbf{v}\|_2^2} + \frac{\left|\mathbf{v}^\dagger \mathbf{u}\right|^2}{\|\mathbf{v}\|_2^2} \tag{2.86}$$

$$= \mathbf{u}^\dagger \mathbf{u} - \frac{\left|\mathbf{v}^\dagger \mathbf{u}\right|^2}{\|\mathbf{v}\|_2^2} \tag{2.87}$$

$$\tag{2.88}$$

and finally

$$\left|\mathbf{v}^\dagger \mathbf{u}\right|^2 \leq \|\mathbf{u}\|_2^2 \|\mathbf{v}\|_2^2, \tag{2.89}$$

$$\left|\mathbf{v}^\dagger \mathbf{u}\right| \leq \|\mathbf{u}\|_2 \|\mathbf{v}\|_2. \tag{2.90}$$

Now we can easily prove the triangle inequality,

$$\|\mathbf{u}+\mathbf{v}\|_2^2 = \mathbf{u}^\dagger\mathbf{u} + \mathbf{u}^\dagger\mathbf{v} + \mathbf{v}^\dagger\mathbf{u} + \mathbf{v}^\dagger\mathbf{v} \tag{2.91}$$

$$= \|\mathbf{u}\|_2^2 + 2\left|\mathbf{u}^\dagger\mathbf{v}\right| + \|\mathbf{v}\|_2^2 \tag{2.92}$$

$$\leq \|\mathbf{u}\|_2^2 + 2\|\mathbf{u}\|\|\mathbf{v}\| + \|\mathbf{v}\|_2^2 \tag{2.93}$$

$$= \left(\|\mathbf{u}\|_2 + \|\mathbf{v}\|_2\right)^2 \tag{2.94}$$

$$\|\mathbf{u}+\mathbf{v}\|_2 \leq \|\mathbf{u}\|_2 + \|\mathbf{v}\|_2. \tag{2.95}$$

Our definition of the 2-norm suggests a generalization,

$$\|\mathbf{v}\|_p = \left(\sum_i |v_i|^p\right)^{1/p}, \tag{2.96}$$

which is called the *p-norm*. This clearly satisfies the non-negativity and scalar multiplication axioms, with proofs analogous to the 2-norm case. Thus we have only to prove the triangle inequality. First, let us define a complementary index q,

$$\frac{1}{p} + \frac{1}{q} = 1 \tag{2.97}$$

which also implies that $p - 1 = \frac{p}{q}$. Now we can show that

$$\|\mathbf{u}+\mathbf{v}\|_p^p = \sum_i |u_i + v_i|^p \tag{2.98}$$

$$\leq \sum_i |u_i||u_i + v_i|^{p-1} + \sum_i |v_i||u_i + v_i|^{p-1} \tag{2.99}$$

$$= \sum_i |u_i||u_i + v_i|^{p/q} + \sum_i |v_i||u_i + v_i|^{p/q} \tag{2.100}$$

Now we will need the *Hölder Inequality* (Kuttler 2012),

$$\sum_i |u_i||v_i| \leq \left(\sum_i |u_i|^p\right)^{1/p} \left(\sum_i |v_i|^q\right)^{1/q} \tag{2.101}$$

which will allow us to bound the two terms on the left

$$\|\mathbf{u}+\mathbf{v}\|_p^p \leq \left(\sum_i |u_i + v_i|^{p/q\cdot q}\right)^{1/q} \left(\sum_i |u_i|^p\right)^{1/p} \tag{2.102}$$

$$+ \left(\sum_i |u_i + v_i|^{p/q\cdot q}\right)^{1/q} \left(\sum_i |v_i|^p\right)^{1/p} \tag{2.103}$$

$$= \left(\|\mathbf{u}+\mathbf{v}\|_p\right)^{p/q} \left(\|\mathbf{u}\|_p + \|\mathbf{v}\|_p\right) \tag{2.104}$$

$$= \left(\|\mathbf{u}+\mathbf{v}\|_p\right)^{p-1} \left(\|\mathbf{u}\|_p + \|\mathbf{v}\|_p\right) \tag{2.105}$$

$$\|\mathbf{u}+\mathbf{v}\|_p \leq \|\mathbf{u}\|_p + \|\mathbf{v}\|_p. \tag{2.106}$$

This reasoning extends to the limit $p \to \infty$ so that

$$\|\mathbf{v}\|_\infty = \max_i |v_i| \tag{2.107}$$

is also a norm called the ∞-norm or max-norm. The unit balls, meaning the set of vector with length less than one, are nicely displayed for different norms in (Trefethen and Bau, III 1997).

In the problems, we will see that any non-singular matrix can induce a norm from some existing norm. With this in mind, we may search for classes of matrices that leave norms invariant, which are sometimes called *isometric* transformations. For example, unitary matrices U leave the 2-norm invariant,

$$\|Uv\|_2^2 = v^\dagger U^\dagger U v = v^\dagger v = \|v\|_2^2. \tag{2.108}$$

If we consider the 1-norm,

$$\|\mathbf{v}\|_1 = \sum_i |v_i| \tag{2.109}$$

and restrict the vector elements to be non-negative, we see that it is invariant under the action of a left stochastic matrix S, as remarked in Chapter 1, since the columns in this matrix have non-negative elements that sum to one,

$$\sum_i s_{ij} = 1. \tag{2.110}$$

Using this property we see that

$$\|S\mathbf{v}\|_1 = \sum_i \left| \sum_j s_{ij} v_j \right| \tag{2.111}$$

$$= \sum_i \sum_j s_{ij} v_j \tag{2.112}$$

$$= \sum_j v_j \tag{2.113}$$

$$= \|\mathbf{v}\|_1 \tag{2.114}$$

The ∞-norm is invariant under the action of a permutation matrix, as are all p-norms, since it just permutes the terms in the sum. Note that a permutation matrix is both unitary and stochastic.

2.6.2 Matrix norms

The simplest idea for a norm on the space of matrices is to treat the matrix as a vector, by unrolling the table of elements into a list, and using a vector norm on this list. This is a vector space under element-wise operations, with a canonical basis $\hat{\mathbf{e}}_{ij}$, and thus we expect this to work. If we think about using the 2-norm

on this vector space, called the *Frobenius norm* on the space of matrices, we may write

$$\begin{aligned} A \cdot B &= \sum_{ij} \bar{A}_{ij} B_{ij} \\ &= \sum_{i} (A^\dagger B)_{ii} \\ &= \mathrm{Tr}\big(A^\dagger B\big), \end{aligned}$$

where was have introduced the *trace* functional over linear operators. For a matrix, the trace is defined as the sum of the diagonal elements, but this definition is invariant under a change of basis. The trace of an operator can be fully characterized, independent of any given basis, by the following three properties,

$$\mathrm{Tr}(A + B) = \mathrm{Tr}(A) + \mathrm{Tr}(B), \tag{2.115}$$

$$\mathrm{Tr}(\alpha A) = \alpha \,\mathrm{Tr}(A), \tag{2.116}$$

$$\mathrm{Tr}(AB) = \mathrm{Tr}(BA), \tag{2.117}$$

In order to prove this, we will again look at the space of matrices as a vector space, with a canonical basis $\hat{\mathbf{e}}_{ij}$, so that an arbitrary matrix A can be written

$$A = \sum_{ij} a_{ij} \hat{\mathbf{e}}_{ij}.$$

Let η be a linear functional on the vector space of square matrices such that

$$\eta(AB) = \eta(BA). \tag{2.118}$$

Then we will prove that η and Tr are proportional. We can start by observing that

$$\eta(AB - BA) = \eta(AB) - \eta(BA) \tag{2.119}$$

$$= \eta(AB) - \eta(AB) \tag{2.120}$$

$$= 0 \tag{2.121}$$

All the terms $\hat{\mathbf{e}}_{ii}$ vanish from $AB - BA$, so we know that

$$\eta(\hat{\mathbf{e}}_{ij}) = \alpha_i \delta_{ij}, \tag{2.122}$$

for some scalar α_i. In addition, we have that for any permutation matrix P,

$$\eta(P^T A P) = \eta(P P^T A) \tag{2.123}$$

$$= \eta(A) \tag{2.124}$$

meaning that the value of the functional does not change under any permutation of i, so that all diagonal terms contribute equally,

$$\eta(\hat{\mathbf{e}}_{ii}) = \eta(\hat{\mathbf{e}}_{00}). \tag{2.125}$$

This means that

$$\eta(A) = \eta(\sum_{ij} A_{ij}\hat{\mathbf{e}}_{ij}) \tag{2.126}$$

$$= \sum_{ij} A_{ij}\eta(\hat{\mathbf{e}}_{ij})\delta_{ij} \tag{2.127}$$

$$= \sum_{i} A_{ii}\eta(\hat{\mathbf{e}}_{ii}) \tag{2.128}$$

$$= \mathrm{Tr}(A)\eta(\hat{\mathbf{e}}_{00}) \tag{2.129}$$

Note that the third property also implies that the trace is the same in any basis since

$$\mathrm{Tr}\{P^{-1}AP\} = \mathrm{Tr}\{APP^{-1}\} = \mathrm{Tr}\{A\}. \tag{2.130}$$

We may also define matrix norms by using vector norms defined on the range space,

$$\|A\| = \max_{\mathbf{v}\neq\mathbf{0}} \frac{\|A\mathbf{v}\|}{\|\mathbf{v}\|}$$

where we note that the norms on the right hand side are vector norms, and they operate potentially in two different spaces. Note that since A is linear, this defnition is invariant to the scaling of $\mathbf{v}$,

$$\frac{\|A\alpha\mathbf{v}\|}{\|\alpha\mathbf{v}\|} = \frac{|\alpha|\|A\mathbf{v}\|}{|\alpha|\|\mathbf{v}\|} = \frac{\|A\mathbf{v}\|}{\|\mathbf{v}\|}$$

so that we can use instead

$$\|A\| = \max_{\|\mathbf{v}\|=1} \|A\mathbf{v}\|. \tag{2.131}$$

This definition is clearly non-negative, and cannot be zero while A has any nonzero columns. It also behaves correctly when multiplying the matrix by a scalar, so we have only to check the triangle inequality. Using the linearity of A and B, and the triangle inequality for the vector norm, we have

$$\|A+B\| = \max_{\|\mathbf{v}\|=1} \|(A+B)\mathbf{v}\| \tag{2.132}$$

$$= \max_{\|\mathbf{v}\|=1} \|A\mathbf{v}+B\mathbf{v}\| \tag{2.133}$$

$$\leq \max_{\|\mathbf{v}\|=1} \|A\mathbf{v}\| + \|B\mathbf{v}\| \tag{2.134}$$

$$\leq \max_{\|\mathbf{v}\|=1} \|A\mathbf{v}\| + \max_{\|\mathbf{v}\|=1} \|B\mathbf{v}\| \tag{2.135}$$

$$= \|A\| + \|B\|. \tag{2.136}$$

These induced matrix norms measure the deformation of the unit ball defined by the vector norm by the transformation defined by the matrix. We can try to

get closed-form expressions for these matrix norms by looking at specific vector norms. For example, the matrix 1-norm is the maximum column sum. We can show this using the definition of matrix multiplication, the triangle inequality,

$$\|A\|_1 = \max_{\|\mathbf{v}\|_1=1} \|A\mathbf{v}\|_1 \tag{2.137}$$

$$= \max_{\|\mathbf{v}\|_1=1} \left\| \sum_j v_j a_j \right\|_1 \tag{2.138}$$

$$\leq \max_{\|\mathbf{v}\|_1=1} \sum_j \|v_j a_j\|_1 \tag{2.139}$$

$$= \max_{\|\mathbf{v}\|_1=1} \sum_j |v_j| \|a_j\|_1 \tag{2.140}$$

$$= \max_j \sum_i |a_{ij}|, \tag{2.141}$$

where we have chosen $\mathbf{v} = \hat{\mathbf{e}}_j$ to maximize the sum. Likewise, the ∞-norm is the maximum row sum,

$$\|A\|_\infty = \max_{\|\mathbf{v}\|_\infty=1} \|A\mathbf{v}\|_\infty \tag{2.142}$$

$$= \max_{\|\mathbf{v}\|_\infty=1} \max_i \left| \sum_j a_{ij} v_j \right| \tag{2.143}$$

$$= \max_i \sum_j |a_{ij}|. \tag{2.144}$$

The 2-norm is more complicated, but we will see in Section 2.8 that it can be characterized using the singular value decomposition.

2.7 Projectors

The definition of the projector is $P^2 = P$ meaning that projecting again does not change the results of the first projection. We can think of a projection as "cutting off" part of a vector, and once the right part is cut off, we do not need to do it again. If P is a projector, then its complement $I - P$ is also a projector

$$(I - P)^2 = I - 2P + P^2 \tag{2.145}$$

$$= I - 2P + P \tag{2.146}$$

$$= I - P, \tag{2.147}$$

and it is called the *complementary projector*.

Remember that the range of an operator is any vector we can make by applying it, or the span of its columns. Thus the range of P is any vector Pv.

The nullspace of P will be any vector of the form $Pv - v$, since

$$P(Pv - v) = P^2v - Pv \tag{2.148}$$
$$= Pv - Pv \tag{2.149}$$
$$= 0. \tag{2.150}$$

Now suppose a vector x was in both range(P) and null(P), then $x = Px$ and $x = x - Px$, so that

$$x - Px = Px \tag{2.151}$$
$$x = 0, \tag{2.152}$$

meaning that

$$\text{range}(P) \cap \text{null}(P) = \emptyset. \tag{2.153}$$

Another way to see this is to look at the complementary projector, and see that

$$\text{range}(I - P) = \text{null}(P). \tag{2.154}$$

The projectors partitions a space into two disjoint subspaces. We can say that our space V is the *direct sum* of range(P) and null(P), written

$$V = \text{range}(P) \oplus \text{null}(P), \tag{2.155}$$

meaning that there is a unique decomposition of any vector $v \in V$ such that

$$v = v_r + v_n \qquad \text{where} \qquad v_r \in \text{range}(P), v_n \text{null}(P). \tag{2.156}$$

Note here that this definition does not mean that vectors in one space are orthogonal to those in the other. For that we need an *orthogonal projector* which satisfies $P = P^\dagger$. If the projection operator is Hermitian, then vectors in the range are orthogonal to those in the nullspace

$$(Pv)^\dagger (I - P)w = x^\dagger P^\dagger (I - P)w \tag{2.157}$$
$$= x^\dagger P(I - P)w \tag{2.158}$$
$$= x^\dagger (P - P^2)w \tag{2.159}$$
$$= x^\dagger (P - P)w \tag{2.160}$$
$$= 0 \tag{2.161}$$

The converse is also true, which is proved in the text by explicitly constructing the SVD of P.

This allows us to see that orthogonal projectors can be represented as $P = QQ^\dagger$ where the columns of Q are orthonormal. Clearly, this representation is still a projector,

$$P^2 = (QQ^\dagger)(QQ^\dagger) = QIQ^\dagger = P.$$

In fact, this can be further analyzed into the sum of rank-one projectors.

$$QQ^\dagger = \sum_i q_i q_i^\dagger$$

since $q_i^\dagger q_j = \delta_{ij}$. We can imagine this in two steps. First the vector is transformed $x \to w$ where $w_i = q_i^\dagger x$. Then the output vector is a sum of the columns weighted by vector entries, $y = \sum_i q_i q_i^\dagger x$, which is exactly how we decomposed vectors in an orthonormal basis.

Oblique projectors are projectors that are not orthogonal. We can write them as

$$uv^\dagger$$

so that the range is span(u) and the nullspace is orthogonal to v. Notice that the range $\mathcal{R}$ and nullspace $\mathcal{N}$ are not orthogonal to each other, but we know the $\dim(\mathcal{R}) = 1$ and $\dim(\mathcal{N}) = n - 1$. We also know that they do not share a vector in common. From above, if $v \in \mathcal{R}$ and $v \in \mathcal{N}$ we have

$$\begin{aligned} v &= v - Pv && \text{(because it is in the nullspace)} \\ &= (I - P)v \\ &= 0 && \text{(because it is in the range)} \end{aligned}$$

Thus the bases for $\mathcal{R}$ and $\mathcal{N}$ form a basis for the whole space. However, how do we know that the resolution of a vector is unique in a non-orthogonal basis? Suppose that a vector v has two different representations $\{\alpha_i\}$ and $\{\beta_i\}$ in the basis. Then $\{\alpha_i - \beta_i\}$ is an expansion for the 0 vector, which is impossible since the basis vectors are linearly independent.

The unitary reflector matrices, or Householder reflectors, are related to projectors, and have the form

$$Q_k = \begin{pmatrix} I_{k-1} & 0 \\ 0 & F \end{pmatrix}$$

so that it leaves the first $k - 1$ rows untouched. The purpose of F is to turn a column a_k into a multiple of the unit vector e_k, meaning put all zeros below the diagonal. Since F will be unitary, the constant must be $\|a_k\|$ so that the norm of the vector does not change. Suppose that we define the vector v between our two points $a_k - \|a_k\| e_k$. If we project onto the space orthogonal to v, using the complementary orthogonal projector

$$I - \frac{vv^\dagger}{v^\dagger v}$$

then we would get the point midway between the two vectors (in 2D, we project onto the perpendicular bisector of v). If we go twice as far, we will have arrived

at the other point. We can show this explicitly,

$$\begin{aligned}\left(I - 2\frac{vv^\dagger}{v^\dagger v}\right) a_k &= a_k - 2\frac{\|a_k\|^2 - \|a_k\|a_{kk}}{v^\dagger v}(a_k - \|a_k\|e_k) \\ &= a_k - 2\frac{\|a_k\|^2 - \|a_k\|a_{kk}}{\|a_k\|^2 - 2\|a_k\|a_{kk} + \|a_k\|^2}(a_k - \|a_k\|e_k) \\ &= a_k - (a_k - \|a_k\|e_k) \\ &= \|a_k\|e_k.\end{aligned}$$

Also, we can prove that this matrix is orthogonal

$$\begin{aligned}\left(I - 2\frac{vv^\dagger}{v^\dagger v}\right)^\dagger \left(I - 2\frac{vv^\dagger}{v^\dagger v}\right) &= \left(I - 2\frac{vv^\dagger}{v^\dagger v}\right)\left(I - 2\frac{vv^\dagger}{v^\dagger v}\right) \\ &= I - 4\frac{vv^\dagger}{v^\dagger v} + 4\frac{vv^\dagger vv^\dagger}{(v^\dagger v)^2} \\ &= I - 4\frac{vv^\dagger}{v^\dagger v} + 4\frac{vv^\dagger}{v^\dagger v} \\ &= I.\end{aligned}$$

We can choose either sign for the nonzero entry in the reflected vector. We should choose the greatest distance from our original vector, or -sign(x_k). These matrices will be important in Grover's Algorithm from Section 4.6.

2.8 The Singular Value Decomposition

2.8.1 Definition of the SVD

The singular value decomposition, or SVD, is a matrix factorization. This means that it takes one matrix as input and produces a set of simpler matrices whose product is equal to the input matrix. There are many factorizations, including LU, QR, and the polar decomposition. The SVD has the form

$$A = USV^\dagger \tag{2.162}$$

where U and V are unitary, and S is diagonal and non-negative. We could guess that a linear map A would send the unit n-sphere to a connected set (since it is continuous) and to a convex set (since it is linear). In fact, A sends the unit ball to an ellipsoid whose axes are determined by U. In the form $AV = US$, it is clear that U is a basis for the range space, and V is a basis for the domain. Multiplying by S on the right scales the columns of U, and thus S tells us the lengths of the axes in the transformed unit ball. Another way to think of the SVD is that it is an incremental, best approximation of A in the 2-norm. The 2-norm is special in that it is unitarily invariant, defined by the inner/dual product, and related to the eigendecomposition.

Now that we have the SVD, we can use it to show that the Frobenius norm of a matrix A is just the sum of the squares of its singular values.

$$\begin{aligned}
\|A\|_F^2 &= \mathrm{Tr}\left(A^\dagger A\right) && (2.163)\\
&= \mathrm{Tr}\left(\left(USV^\dagger\right)^\dagger USV^\dagger\right) && (2.164)\\
&= \mathrm{Tr}\left(VS^\dagger U^\dagger USV^\dagger\right) && (2.165)\\
&= \mathrm{Tr}\left(VS^\dagger SV^\dagger\right) && (2.166)\\
&= \mathrm{Tr}\left(S^\dagger SV^\dagger V\right) && (2.167)\\
&= \mathrm{Tr}\left(SS\right) && (2.168)\\
&= \sum_i \sigma_i^2 && (2.169)
\end{aligned}$$

We can also prove that the absolute value of the determinant of a square matrix is the product of the singular values,

$$\begin{aligned}
|\det(A)| &= \left|\det\left(USV^\dagger\right)\right| && (2.170)\\
&= |\det(U)||\det(S)|\left|\det\left(V^\dagger\right)\right| && (2.171)\\
&= |\det(S)| && (2.172)\\
&= \prod_i \sigma_i && (2.173)
\end{aligned}$$

and the trace is bounded by the sum of the singular values.

$$\begin{aligned}
|\mathrm{Tr}(A)| &= \left|\mathrm{Tr}\left(USV^\dagger\right)\right| && (2.174)\\
&= \left|\mathrm{Tr}\left(SV^\dagger U\right)\right| && (2.175)\\
&= |\mathrm{Tr}(SW)| && (2.176)\\
&= \left|\sum_i \sigma_i w_{ii}\right| && (2.177)\\
&\leq \sum_i |\sigma_i w_{ii}| && (2.178)\\
&\leq \sum_i \sigma_i && (2.179)
\end{aligned}$$

The last line follows from the fact that

$$\begin{aligned}
|w_{ii}| &= \left|\mathbf{v}_i^\dagger \mathbf{u}_i\right| && (2.180)\\
&\leq \|\mathbf{v}_i\|\|\mathbf{u}_i\| && (2.181)\\
&= 1 && (2.182)
\end{aligned}$$

In fact, all unitary matrices have elements with absolute value less than unity, since the 2-norm of each column is unity. We will see alternate representations of the determinant and trace using the eigendecomposition in Section 2.9

Finally, we can use the SVD to solve a system of linear equations

$$Ax = b.$$

We first factor the system matrix A using the SVD, and then invert the component matrices one-by-one since each has a simple inverse,

$$\begin{aligned} USV^\dagger x &= b \\ SV^\dagger x &= U^\dagger b \\ V^\dagger x &= S^{-1}U^\dagger b \\ x &= VS^{-1}U^\dagger b \end{aligned}$$

This strategy can also be used to solve a least-squares problem

$$\min \|b - Ax\|_2 = \min \left\|b - USV^\dagger x\right\|_2 = \min \left\|U^\dagger b - SV^\dagger x\right\|_2.$$

We will use the full SVD, so that for a rank r matrix, the bottom $n - r$ rows of the second term cannot be cancelled. Thus

$$(U^\dagger b)_i \text{ for } i \geq r$$

is the residual of our least-squares solution. If this is zero, we say that the rhs b is *consistent* with the system A. Looking at the first r rows, which we label "top", we have

$$\begin{aligned} (U^\dagger b)_{top} &= SV^\dagger x \\ x &= VS^{-1}(U^\dagger b)_{top}. \end{aligned}$$

2.8.2 Proof that the SVD exists

Most factorizations are justified using constructive arguments which proceed inductively, making the factorization one column or row at a time. This is exactly how we will proceed for the SVD. We start by setting

$$\sigma_1 = \|A\|_2.$$

Remember that this value arises from taking a maximum over a continuous set of directions, we repeat Eq. (2.131),

$$\|A\|_2 = \max_{\|\mathbf{v}\|=1} \|A\mathbf{v}\|_2.$$

A "compactness argument" tells us that the maximizing vector $\mathbf{v}$ actually exists in the vector space, so that $A\mathbf{v}_1 = \sigma_1 \mathbf{u}_1$ with $\|\mathbf{v}_1\| = \|\mathbf{u}_1\| = 1$. We will go through this in detail since it is usually omitted. The compactness argument is that the unit sphere in $\mathbb{R}^n$ and $\mathbb{C}^n$ is compact, and thus a continuous function on it attains in the space its maximum (and minimum) over the unit sphere. The unit sphere is compact because it is closed and bounded. It is closed because

it is the complement of an open set, all the points not on a sphere. That set is open because I can draw a ball around any point in it without intersecting the sphere. The vector norm is continuous because that is how we understand normed spaces, namely that the norm maps an open set of vectors into an open interval of real numbers. Starting with these vectors, $\mathbf{v}_1$ and $\mathbf{u}_1$, we can build orthonormal bases for both the domain and range spaces. Let V_1 and U_1 be unitary matrices representing these bases. This means that we have

$$U_1^\dagger A V_1 = S_1 = \begin{pmatrix} \sigma_1 & \mathbf{w}_1^\dagger \\ \mathbf{0} & B \end{pmatrix}$$

where $\mathbf{0}$ has dimension $m-1$, $\mathbf{w}_1$ dimension $n-1$, and B dimension $(m-1)\times(n-1)$.

Next, we use properties of the 2-norm to show that our division is, in fact, block diagonal. Let us define a vector

$$\mathbf{w} = \begin{pmatrix} \sigma_1 \\ \mathbf{w}_1 \end{pmatrix},$$

and act with S_1 and take the norm,

$$\begin{aligned} \|S_1\mathbf{w}\|_2 &= \left\| \begin{pmatrix} \sigma_1 & \mathbf{w}_1^\dagger \\ \mathbf{0} & B \end{pmatrix} \begin{pmatrix} \sigma_1 \\ \mathbf{w}_1 \end{pmatrix} \right\|_2 \\ &= \left\| \begin{pmatrix} \sigma_1^2 + \mathbf{w}_1^\dagger \mathbf{w}_1 \\ B\mathbf{w}_1 \end{pmatrix} \right\|_2 \\ &\geq \sigma_1^2 + \mathbf{w}_1^\dagger \mathbf{w}_1 \\ &= \sqrt{\sigma_1^2 + \|\mathbf{w}_1\|_2^2} \left\| \begin{pmatrix} \sigma_1 \\ \mathbf{w}_1 \end{pmatrix} \right\|_2 \end{aligned}$$

The inequality gives us a bound on the norm of S_1, using again Eq. (2.131), namely

$$\|S_1\|_2 \geq \sqrt{\sigma_1^2 + \|\mathbf{w}_1\|_2^2}.$$

However, since U_1 and V_1 are unitary, we know that

$$\begin{aligned} \|S_1\|_2 &= \left\| U_1^\dagger A V_1 \right\|_2 \\ &= \|A\|_2 \\ &= \sigma_1. \end{aligned}$$

This, together with Eq. (2.78), implies that $\mathbf{w}_1 = \mathbf{0}$.

Lastly, we recurse on B. If B has nonzero dimension, we can employ the same factorization for B,

$$U_2^\dagger B V_2 = S_2.$$

This allow us to telescope the factorization

$$\begin{aligned} A &= U_1 \begin{pmatrix} 1 & 0 \\ 0 & U_2 \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & V_2^\dagger \end{pmatrix} V_1^\dagger, \\ &= U_1 \begin{pmatrix} 1 & 0 \\ 0 & U_2 \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & \mathbf{w}_2^\dagger \\ 0 & 0 & C \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & V_2^\dagger \end{pmatrix} V_1^\dagger. \end{aligned}$$

We can keep recursing until the matrix in the lower right vanishes. The product of unitary matrices on the left and right can be written as a single unitary matrix.

We can understand this sequence of operations as generating better and better approximations of the matrix A. This notion can be made precise by looking at the accuracy of the approximation. For ν in $0 \leq \nu \leq r$ where $\text{rank}(A) = r$, define

$$A_\nu = \sum_{i=1}^{\nu} \sigma_i u_i v_i^\dagger, \tag{2.183}$$

where $\{u_i\}$ are the left singular vectors (columns of U), $\{v_i\}$ are the right singular vectors (columns of V), and $\sigma_{r+1} = 0$. Now we can compute the approximation error

$$\|A - A_\nu\|_2 = \inf_{\text{rank}(B) \leq \nu} \|A - B\|_2 = \sigma_{\nu+1}. \tag{2.184}$$

Suppose, in contradiction to our hypothesis, that there exists a matrix B which better approximates A, namely

$$\|A - B\|_2 < \|A - A_\nu\|_2 = \sigma_{\nu+1}.$$

If $\text{rank}(B) \leq \nu$, then there must exist an $(n-\nu)$-dimensional subspace W such that

$$B\mathbf{w} = 0 \quad \forall \mathbf{w} \in W.$$

We can then bound the norm of the action of A on any vector from W,

$$\begin{aligned} \|A\mathbf{w}\|_2 &= \|(A - B)\mathbf{w}\|_2 \\ &\leq \|A - B\|_2 \|\mathbf{w}\|_2 \\ &< \sigma_{\nu+1} \|\mathbf{w}\|_2 \end{aligned}$$

In sum, W is an $(n-\nu)$-dimensional subspace in which $\|A\mathbf{w}\|_2 < \sigma_{\nu+1}\|\mathbf{w}\|_2$. However, there is an $(\nu+1)$-dimensional subspace in which $\|A\mathbf{v}\|_2 \geq \sigma_{\nu+1}\|\mathbf{v}\|_2$, namely the space spanned by the first $(\nu+1)$ right singular vectors. Since the sum of the dimensions exceeds n, there must be some vector lying in the intersection, which contradicts the norm bounds, and thus no better approximation B can exist.

$$\|A - A_\nu\|_F = \inf_{\text{rank}(B) \leq \nu} \|A - B\|_F = \sqrt{\sigma_{\nu+1}^2 + \cdots + \sigma_r^2}. \tag{2.185}$$

2.8.3 Bases and Unitary Operators

If I have an orthogonal basis $\{b_i\}$, it means that $b_i^\dagger b_j = 0$ if $i \neq j$. If in addition the basis is orthonormal, then we have the more compact expression $b_i^\dagger b_j = \delta_{ij}$. Now suppose we create a matrix B whose columns are the basis vectors,

$$B = (b_0|b_1|\dots|b_n)\,.$$

Since this is a basis, we will have n vectors for an n-dimensional space, and B is square. If we look at the matrix product $B^\dagger B$,

$$\begin{aligned}(B^\dagger B)_{ij} &= \sum_k (B^\dagger)_{ik}(B)_{kj} \\ &= \sum_k \overline{(B)}_{ki}(B)_{kj} \\ &= \sum_k \overline{(b_i)}_k (b_j)_k \\ &= b_i^\dagger b_j \\ &= \delta_{ij}\end{aligned}$$

we see that $B^\dagger B = I$. Note that this also implies $BB^\dagger = I$ by conjugating both sides. Thus I may think of any unitary operator as an orthonormal basis, and vice versa.

We can express the vector x in the U basis, meaning get the set of coefficients for the U basis vectors, using $U^\dagger x$ since the ith component in the u-basis is the inner

$$x = \sum_j x_j u_j$$

so that

$$u_i^\dagger x = u_i^\dagger \sum_j x_j u_j \tag{2.186}$$

$$= \sum_j x_j u_i^\dagger u_j \tag{2.187}$$

$$= \sum_j x_j \delta_{ij} \tag{2.188}$$

$$= x_i. \tag{2.189}$$

Using the same reasoning, the change of basis for an operator A is $U^\dagger AU$. Note that this is the root of the two views of quantum mechanics. We can see the wave function changing, or the operators changing.

2.9 Eigenproblems

The eigendecomposition of a matrix, when it exists, is given by

$$X^{-1}AX = \Lambda \tag{2.190}$$

where X is non-singular and Λ is diagonal.

We can use the eigendecomposition to show that the determinant is the product of the eigenvalues,

$$\begin{aligned}
\det(A) &= \det\big(X\Lambda X^{-1}\big) && (2.191)\\
&= \det\big(\Lambda X^{-1}X\big) && (2.192)\\
&= \det(\Lambda) && (2.193)\\
&= \prod_i \lambda_i. && (2.194)
\end{aligned}$$

Likewise, the trace of a matrix is the sum of its eigenvalues,

$$\begin{aligned}
\mathrm{Tr}(A) &= \mathrm{Tr}\big(X\Lambda X^{-1}\big) && (2.195)\\
&= \mathrm{Tr}\big(\Lambda X^{-1}X\big) && (2.196)\\
&= \mathrm{Tr}(\Lambda) && (2.197)\\
&= \sum_i \lambda_i. && (2.198)
\end{aligned}$$

This also shows that both the determinant and trace are invariants, since they do not depend on the basis used to define the matrix.

2.10 Problems

Problem II.1 This problem will familiarize you with the grading system that we use at UB. Follow the steps below to ensure that your Autolab account is working correctly.

1. Create your account at https://autograder.cse.buffalo.edu using your UB email address.

2. An account may have been created for you if you enrolled before you had an account If Autolab says that you already have an account, click "Forgot your password?" and enter your email address. Follow instructions to reset your password.

3. Ensure that you are registered for the course: CSE410: Quantum Algorithms (Fall 19)

4. Submit a pdf to Homework 0 with the following information:

   - Name

- Person number
- The equation

$$\frac{x^T Ax}{||x||^2} \to \lambda_{\max}$$

since equation writing will be essential in this course.

The best way to create PDF from LaTeX is to use `pdflatex`,

```
pdflatex essay.tex
bibtex essay
pdflatex essay.tex
pdflatex essay.tex
```

where the repetition is necessary to assure that the metadata stored in auxiliary files is consistent. This process can be handled in an elegant way by using the `latexmk` program,

```
latexmk -pdf essay.tex
```

If you rely on TeX source or BibTeX files in other locations, you can use

```
TEXINPUTS=${TEXINPUTS}:/path/to/tex BIBINPUTS=${BIBINPUTS}:/path/to/bib
  latexmk -pdf essay.tex
```

Problem II.2 Show that the original Vandermonde matrix, defined by

$$V = \begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \tag{2.199}$$

is actually a change of basis from the monomial basis $\{x^k\}$ to the basis of point evaluation functionals $\{\eta_{x_i}\}$

$$\eta_z(\phi) = \int \phi(x)\delta(x-z)dx \tag{2.200}$$

Problem II.3 Implement both Classical and Modified Gram-Schmidt orthogonalization in PETSc. Use an example to show instability in the classical algorithm that is not present in the modified form.

Problem II.4 NLA 1.1

Problem II.5 NLA 1.3

Problem II.6 NLA 1.4

Problem II.7 NLA 2.1

Problem II.8 NLA 2.2

Problem II.9 NLA 2.3

Problem II.10 NLA 2.4

Problem II.11 NLA 2.6

Problem II.12 NLA 2.7

Problem II.13 NLA 3.1

Problem II.14 NLA 3.3

Problem II.15 NLA 3.6

Problem II.16 NLA 4.1

Problem II.17 NLA 4.4

Problem II.18 NLA 5.3

Problem II.19 NLA 5.4

Problem II.20 QALA 3.1

Problem II.21 QALA 3.4

Problem II.22 QALA 3.5

Problem II.23 QALA 3.7

Problem II.24 QALA 3.9

Problem II.25 QALA 3.10

Problem II.26 QALA 3.12

Problem II.27 QALA 3.16

Problem II.28 QALA 3.17

Problem II.29 NLA 6.1

Problem II.30 NLA 6.3

Problem II.31 NLA 6.5

Problem II.32 NLA 7.1

Problem II.33 NLA 7.3

Problem II.34 NLA 7.4

Problem II.35 NLA 12.2

Problem II.36 NLA 14.1

Problem II.37 NLA 15.2

References

Fearnley-Sander, Desmond (1979). "Hermann Grassmann and the Creation of Linear Algebra". In: *The American Mathematical Monthly* 86, pp. 809–817. URL: http://www.maa.org/sites/default/files/pdf/upload_library/22/Ford/DesmondFearnleySander.pdf.

Wikipedia (2015). *Linear Algebra*. https://en.wikipedia.org/wiki/Linear_algebra. URL: https://en.wikipedia.org/wiki/Linear_algebra.

Lawson, C. L., R. J. Hanson, D. Kincaid, and F. T. Krogh (1979). "Basic linear algebra subprograms for Fortran usage". In: *ACM Transactions on Mathematical Software* 5, pp. 308–323.

Anderson, E., Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen (May 1990). *LAPACK: A portable linear algebra library for high-performance computers*. Tech. rep. CS-90-105. Computer Science Dept., University of Tennessee.

Falgout, R. (2017). *hypre Users Manual*. Tech. rep. Revision 2.11.2. Lawrence Livermore National Laboratory.

— (n.d.). *hypre Web page*. http://www.llnl.gov/CASC/hypre.

Heroux, Michael A. and James M. Willenbring (2003). *Trilinos Users Guide*. Tech. rep. SAND2003-2952. Sandia National Laboratories. URL: http://trilinos.sandia.gov/.

Heroux et al., M. (n.d.). *Trilinos Web page*. `http://trilinos.sandia.gov/`.

Bastian, Peter, Markus Blatt, Andreas Dedner, Christian Engwer, Jorrit Fahlke, Christoph Gersbacher, Carsten Gräser, Christoph Grüninger Robert Klöfkorn, Steffen Müthing, Martin Nolte, Mario Ohlberger, and Oliver Sander (2015). *DUNE Web page*. http://www.dune-project.org/. URL: http://www.dune-project.org/.

Jacob, Benoit and Gaël Guennebaud (2015). *Eigen Web page*. http://eigen.tuxfamily.org/. URL: http://eigen.tuxfamily.org/.

Poulson, Jack, Bryan Marker, Jeff R. Hammond, Nichols A. Romero, and Robert van de Geijn (2013). "Elemental: A New Framework for Distributed Memory Dense Matrix Computations". In: *ACM Transactions on Mathematical Software* 39.2.

Poulson, Jack (2015). *Elemental: Distributed memory dense linear algebra*. http://libelemental.org. URL: http://libelemental.org/.

Balay, Satish, Shrirang Abhyankar, et al. (2020). *PETSc Users Manual*. Tech. rep. ANL-95/11 - Revision 3.14. Argonne National Laboratory.

— (2019). *PETSc Web page*. https://www.mcs.anl.gov/petsc. URL: https://www.mcs.anl.gov/petsc.

Balay, Satish, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith (1997). "Efficient Management of Parallelism in Object Oriented Numerical Software Libraries". In: *Modern Software Tools in Scientific Computing*. Ed. by E. Arge, A. M. Bruaset, and H. P. Langtangen. Birkhauser Press, pp. 163–202.

Knepley, Matthew G. (2012). "Programming Languages for Scientific Computing". In: *Encyclopedia of Applied and Computational Mathematics*. Ed. by Björn Engquist. Springer. DOI: 10.1007/978-3-540-70529-1. URL: http://arxiv.org/abs/1209.1711.

Kuttler, Kenneth (2012). *Linear algebra: theory and applications*. URL: http://www.math.byu.edu/~klkuttle/Linearalgebra.pdf.

Trefethen, Lloyd N. and David Bau, III (1997). *Numerical Linear Algebra*. Philadelphia, PA: Society for Industrial and Applied Mathematics, pp. xii + 361. ISBN: 0-89871-361-7.

Chapter 3

Boolean and Hilbert Spaces

Science does not advance by derivation, but mainly by intuition and inspired guessing. Derivation is very useful to become certain of a result, but it rarely is the impetus. Thus mental models are incredibly important when thinking about new scientific phenomena. Our mental model of quantum mechanics will greatly influence our ability to think about and create new quantum algorithms. Therefore, we will examine some differences between our mental models of classical and quantum mechanics. In an insightful paper (Hardy 2001), Hardy lays out the fundamental distinction between the classical and quantum settings.

What is a state? A *state* is something with enough information to determine all possible (probabilistic) measurements. It is specified by K numbers, where K is called the number of degrees of freedom.

What is the dimension? The *dimension* is the number of states that can be reliably distinguished from each other by a *one-shot experiment* or measurement. Here we are using "one-shot" to distinguish a single measurement from a series of measurements used to build up a probability distribution. For example, we can tell the difference between a head or tail with a single measurement, or between horizontal and vertical polarization of a photon, but I could not tell whether the photon is polarized at a slant even though this is a valid state of the quantum system.

What is a pure state? A *pure state* is an extremal point in the state space, namely one that cannot be made as a convex combination of other states. These represent *definite* states of the system, meaning those that cannot be composed of statistical combinations of other states. It seems to be a fundamental difference between classical and quantum mechanics that there are an finite number of pure states in the former. An excellent discussion of classical and quantum pure states appears in (Mallesh et al. 2012).

What is a mixed state? A mixed state is a statistical combination of pure states. Another way of looking at this is that a mixed state is made from a pure state by taking the partial trace over the part of the system that we do not know about. A central difference between classical and quantum spaces is that there are K pure states for $K = N$ degrees of freedom in a classical system, whereas a quantum system can support an infinite number of pure states and has $K = N^2$ degrees of freedom. Having dimension N means that we have N basis vectors in our space, classical or quantum. In fact, this is why we use complex Hilbert spaces to describe states in quantum mechanics, since they have the right relation between the dimension and number of states.

Since we will always construct our composite systems from a combination of two-state systems, our full system will be a tensor product and $N = 2^n$ where n is the number of two-state systems. This also means that we can make a correspondence between n-bit Boolean strings and basis vectors. For example, a basis vector of the full system $\hat{\mathbf{e}}_M$ where the number M can be expressed by the bit string,

$$M = m_0 m_1 \dots m_n,$$

is given by the product of basis vectors of the two-state subsystems

$$\hat{\mathbf{e}}_M = \hat{\mathbf{e}}_{m_0} \otimes \hat{\mathbf{e}}_{m_1} \otimes \cdots \otimes \hat{\mathbf{e}}_{m_n}.$$

We will call a two-state quantum system a *qubit*, or quantum bit, since a two-state classical system is known as a *bit*. We can use the density matrix formalism to model incoherent superpositions of qubits, meaning statistical ensembles of qubits. However, measurements of qubits are inherently probabilistic, even for pure states, and thus it seem like to correct classical analogy would be a probabilistic classical two-state system, which we will call a *probit*. Notice, however, that qubits and probits are quite different. Qubits need not be part of statistical ensembles, and the probabilities between measurements can interfere and cancel, which cannot happen with either a probit or with the probabilities arising from statistical collections of qubits.

Quantum computers are machines for manipulating qubits. A qubit is the basic unit of quantum information. A qubit is a two-state quantum mechanical system, such as the spin of the electron in which the two states can be taken as spin up and spin down, or the polarization of a single photon in which the two states can be taken to be the vertical polarization and the horizontal polarization. In a classical system, a bit would have to be either true or false. However, quantum mechanics allows the qubit to be in a coherent superposition of both states at the same time, a property that is fundamental to quantum mechanics and thus quantum computing. In terms of probabilities, as pointed out by Hardy, quantum amplitudes can be negative, and lead to cancellation, whereas classical probabilities must be positive. Therefore a qubit corresponds to a line in a quantum circuit diagram, but not to a row of the permutation matrix representing our invertible function F, introduced in what follows. The state space for a full problem is a tensor product of individual spaces for each

qubit. When quantum mechanics refer to a *superposition* of states, what they mean is that we have a linear combination of tensor product basis vectors.

Evolution of a quantum system is expressed by the action of a unitary operator. This is so because we want the total probability to be conserved, so that the probability of something happening is always unity,

$$\|\psi(t)\| = \|U_t\psi(0)\| = \|\psi(0)\|$$

and the 2-norm of $\psi(0)$ is the probability of beginning in any state. In the case of probits, evolution using stochastic matrices preserves the total probability.

3.1 Boolean Functions

In order to talk about two-state quantum systems, we will use the language of boolean functions, where we identify the two quantum states with T and F. A *unary Boolean function* operates on a single bit and returns a single bit. There are only two unary functions, NOT and the identity. A *binary Boolean function* operates on two input bits and returns a single output bit. These are the familiar functions, such as AND and OR. We can generalize Boolean functions to strings of bits in at least two different ways. A *bitwise Boolean function* applies a binary function to each pair of bits and collects the output bits into another string, whereas an *n-ary Boolean function* operates on all argument bits to produce a single output bit. For example, n-ary AND returns T only if all input bits are T, and n-ary OR returns F only if all inputs are F. The n-ary XOR function returns T only if an odd number of argument bits are T, which makes sense given its identification with addition modulo 2. We can define the *Boolean inner product* of two bit strings as n-ary XOR of the bitwise AND of the two input strings,

$$x \bullet y = x_1 y_1 \oplus \cdots \oplus x_m y_m. \tag{3.1}$$

This makes some sense in that n-ary XOR looks like addition and bitwise AND looks like multiplication, and we retain the distributive property,

x	y	z	$x \wedge (y \oplus z)$	$(x \wedge y) \oplus (x \wedge z)$
T	T	T	F	F
T	T	F	T	T
T	F	T	T	T
T	F	F	F	F
F	T	T	F	F
F	T	F	F	F
F	F	T	F	F
F	F	F	F	F

Note that n-ary XOR is addition mod 2 of the input bits, rather than addition mod 2 of the input numbers represented by the bit strings. The table for $x \oplus y$ where x and y are single bits is given by

y \ x	0	1
0	0	1
1	1	0

If we instead look at two bit strings,

y \ x	00	01	10	11
00	0	1	0	1
01	1	0	1	0
10	0	1	0	1
11	1	0	1	0

we have copies of the first table, because only the least significant bits matter in addition modulo two.

3.2 Matrix Representations

If we imagine a system composed of n two-state quantum systems, the size of the overall Hilbert space for the combined system is $N = 2^n$, because it is the tensor product of two-dimensional spaces. We can label each basis function of the combined system by its number in binary, so the rightmost bit is fastest. For example, if we combine two electrons, we have states 00, 01, 10, and 11 where 0 and 1 correspond to spin up and spin down basis states.

In quantum mechanics, linear operators transform basis states into each other, and in fact we require that the operators be unitary in closed systems so that the total probability for all measurements remains unity. This mapping can be seen as a transformation of truth values from input to output. However, if we want to represent unitary mappings, we must use invertible Boolean functions. The Boolean function $f(x_1, \ldots, x_n) = y$ is not invertible, so instead we create an invertible function F from it

$$F(x_1, \ldots, x_n, z) = (x_1, \ldots, x_n, z \oplus f(x_1, \ldots, x_n)) \tag{3.2}$$

which can be shown to be its own inverse

$$\begin{aligned} F\left(F(x_1, \ldots, x_n, z)\right) &= F\left(x_1, \ldots, x_n, z \oplus f(x_1, \ldots, x_n)\right) \\ &= (x_1, \ldots, x_n, (z \oplus f(x_1, \ldots, x_n)) \oplus f(x_1, \ldots, x_n)) \\ &= (x_1, \ldots, x_n, z)\,. \end{aligned}$$

As a simple example, the unary NOT function is invertible, and has the representation

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{3.3}$$

We can check that it is invertible (and unitary) by noting that $X^2 = I$. Now suppose we use our strategy for making invertible Boolean functions on the

identity function so that we have $f(x) = x$, the we use $F(x_1, x_2) = (x_1, x_2 \oplus x_1)$. Then we have the matrix representation

$$\begin{array}{c} \\ e_{00} \\ e_{01} \\ e_{10} \\ e_{11} \end{array} \begin{array}{c} \begin{array}{cccc} e_{00} & e_{01} & e_{10} & e_{11} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array} = \begin{pmatrix} I & 0 \\ 0 & X \end{pmatrix}. \tag{3.4}$$

We will call this operation **CNOT** (controlled NOT), since it negates the second argument if and only if the first argument is T. Thus the first argument is *controlling* the NOT on the second. We also note that this matrix cannot be written as a Kronecker product of simpler gates.

In general, the matrix associated with F is size $2N \times 2N$, since we add an extra argument z. We can label each row by the input $x_1 x_2 \cdots x_n z$, and each one will have only a single nonzero in column $x_1 x_2 \cdots x_n (z \oplus f(x_1, \ldots, x_n))$. A matrix with this structure is a *permutation matrix*, which we denote $\boldsymbol{P}_f$. Note that permutation matrices are unitary, which implies the invertibility of F. We can keep going in this fashion by making extra inputs $z_1, \ldots, z_m$ if the function f has m outputs.

We can repeat the same invertibility trick for the AND function $f(x, y) = xy$, since conjunction is equivalent to bit multiplication. We use $F(x_1, x_2, x_3) = (x_1, x_2, x_3 \oplus (x_1 x_2))$, so that the matrix representation is given by

$$\begin{array}{c} \\ e_{000} \\ e_{001} \\ e_{010} \\ e_{011} \\ e_{100} \\ e_{101} \\ e_{110} \\ e_{111} \end{array} \begin{array}{c} \begin{array}{cccccccc} e_{000} & e_{001} & e_{010} & e_{011} & e_{100} & e_{101} & e_{110} & e_{111} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array} = \begin{pmatrix} I_6 & 0 \\ 0 & X \end{pmatrix}. \tag{3.5}$$

This is known as the *Toffoli gate*, and is similar to **CNOT** except the first two inputs are controls. It turns out that we can make any Boolean function out of Toffoli gates since we can emulate both NOT and AND gates,

$$\mathbf{NOT}(x) = \mathbf{TOF}(1, 1, x) \tag{3.6}$$

$$\mathbf{AND}(x, y) = \mathbf{TOF}(x, y, 0) \tag{3.7}$$

It is shown in (Lipton and Regan 2014) that only a polynomial number of ancilla are needed for any circuit, so the Toffoli gate can be considered universal for quantum computation.

We say a Boolean function is *feasible* if the circuit which computes it has polynomial size. We would like the same idea for quantum computing, but as Reagan and Lipton note, the reasoning is more slippery. For now we will make due with some Rules for Feasibility:

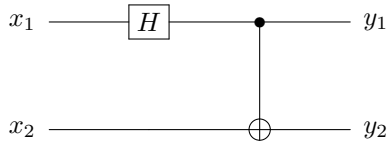
1. Any unitary operator $\boldsymbol{B}$ of size 2^k for fixed k is feasible. These are operations involving a fixed number k of qubits.

2. A tensor product of $\boldsymbol{B}$ with identity matrices is feasible. We will call this a *basic operator*. Note that this is also unitary.

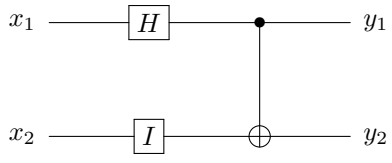
3. The multiplication $\boldsymbol{U}_1 \cdots \boldsymbol{U}_t$ of a polynomial number of feasible operators, so that $t = n^{\mathcal{O}(1)}$, is feasible. We can generalize this to allow $s = n^{\mathcal{O}(1)}$ qubits instead of just n qubits.

We note here that there are exponentially many diagonal matrices, but only polynomially many of them can be feasible, so there are a lot of infeasible diagonal matrices.

We now have at least three separate representations of the operations used in quantum algorithms. First, the quantum circuit picture in terms of quantum wires and gates; second, the linear algebraic representation in the tensor basis of all qubits; and third the Boolean function representation $F(x_1, \ldots, x_n, z_1, \ldots, z_m)$. Let's look at a simple quantum circuit, in order to look at the three different presentations,



which acts on qubit 1 with a Hadamard gate and then feeds both qubits into a **CNOT**. There is an implied identity transformation on qubit 2, which could be included explicitly



in order to make the transition to linear algebra clearer. We can get out linear

algebraic form U for this circuit

$$U = U_2 U_1 \tag{3.8}$$
$$= \mathbf{CNOT}\,(\mathbf{H} \otimes \mathbf{I}) \tag{3.9}$$
$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \left(\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right) \tag{3.10}$$
$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \tag{3.11}$$
$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{pmatrix} \tag{3.12}$$
$$\tag{3.13}$$

Suppose that we act on the input state e_{00} which means that both qubits are in the spin up or F state,

$$Ue_{00} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.14}$$
$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{3.15}$$
$$= \frac{1}{\sqrt{2}} \left(e_{00} + e_{11} \right). \tag{3.16}$$

The output state is the so-called *Bell state*, meaning a maximally entangled state, since if I measure the first qubit and get 0 I know immediately that the other quibit must be 0, and likewise with 1. Thus the circuit above is routinely used to construct an entangled pair from a simple initial state.

In the Boolean function representation, we have the invertible function for **CNOT**,

$$F_{CNOT}(x_1, x_2) = (x_1, x_1 \oplus x_2)$$

so how do we use this? We should think of this as acting on the *index* of the input vector. For example, if we give $\hat{\mathbf{e}}_{10}$ to this gate, then we get $\hat{\mathbf{e}}_{11}$ out. For an arbitrary input, we would have

$$F_{CNOT}\hat{\mathbf{e}}_{ij} = \hat{\mathbf{e}}_{i,i\oplus j}$$

and for a general F

$$F\hat{\mathbf{e}}_{i_1 i_2 \cdots i_n j_1 \cdots j_m} = \hat{\mathbf{e}}_{i_1 \cdots i_n, j_1 \oplus f_1(i_1,\ldots,i_n), \cdots, j_m \oplus f_m(i_1,\ldots,i_n)}. \tag{3.17}$$

We do not have a Boolean formula representation of $\mathbf{H}$, but we can express the operation on indices just as we can with F. In our example $\mathbf{H}$ operates on only the first index of the input. Putting everything together for this circuit, we have

$$\begin{aligned} F_{CNOT}\left(\mathbf{H}\otimes\mathbf{I}\right)\hat{\mathbf{e}}_{00} &= F_{CNOT}\frac{1}{\sqrt{2}}\left(e_{00}+e_{10}\right) && (3.18)\\ &= \frac{1}{\sqrt{2}}\left(F_{CNOT}e_{00}+F_{CNOT}e_{10}\right) && (3.19)\\ &= \frac{1}{\sqrt{2}}\left(e_{00}+e_{11}\right) && (3.20) \end{aligned}$$

which gives us the same Bell state on output. This Boolean function notation is noticeably more compact than the linear algebraic notation, and one of the key innovations by Reagan and Lipton.

3.3 Hadamard Matrices

We define the smallest Hadamard matrix H_2

$$H_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{3.21}$$

and then recursively define all Hadamard matrices with power-of-two size

$$H_N = H_{N/2}\otimes H_2 = \begin{pmatrix} H_{N/2} & H_{N/2} \\ H_{N/2} & -H_{N/2} \end{pmatrix}. \tag{3.22}$$

We can see that H_N is unitary based upon the unitarity of $H_{N/2}$ and the form above, or by noting that $H_N = H_2\otimes\cdots\otimes H_2$ with n copies of H_2 and that tensor products of unitary matrices are unitary. In fact, we can obtain an expression of any entry of H_N

$$H_{ij}^N = \frac{1}{\sqrt{N}}(-1)^{i\bullet j} \tag{3.23}$$

Let's verify this formula first for H_2. In this case, i and j are single bits, and thus only one when they are both one. Since there is only a negative sign in $(1,1)$ entry, the formula is correct. Now for H_4, there is a copy of H_2 for the

high order bit, with a negation when both high order bits are 1. Thus we have

$$\begin{aligned} H^4_{ij} &= \frac{1}{\sqrt{2}}(-1)^{i_1 \wedge j_1} H^2_{i_2 j_2} \\ &= \frac{1}{\sqrt{4}}(-)1^{i_1 \wedge j_1}(-1)^{i_2 \wedge j_2} \\ &= \frac{1}{\sqrt{4}}(-1)^{(i_1 \wedge j_1) \oplus (i_2 \wedge j_2)} \\ &= \frac{1}{\sqrt{4}}(-1)^{i \cdot j}. \end{aligned}$$

We can proceed by induction, where $i' = i_2 \dots i_N$,

$$\begin{aligned} H^N_{ij} &= \frac{1}{\sqrt{2}}(-1)^{i_1 \wedge j_1} H^{N-1}_{i'j'} \\ &= \frac{1}{\sqrt{4}}(-1)^{i_1 \wedge j_1}(-1)^{i' \cdot j'} \\ &= \frac{1}{\sqrt{4}}(-1)^{(i_1 \wedge j_1) \oplus i' \cdot j'} \\ &= \frac{1}{\sqrt{4}}(-1)^{i \cdot j}. \end{aligned}$$

Using this expression, the action of H on a vector x is given by

$$y_i = \sum_j H^N_{ij} x_j \tag{3.24}$$

$$= \frac{1}{\sqrt{N}} \sum_j (-1)^{i \cdot j} x_j. \tag{3.25}$$

3.4 Measuring Entanglement

The simplest measure of entanglement is the determination that two states are indeed entangled. For qubits this means that that the four dimensional product state is not factorizable into the tensor product of two qubit states. Let's look at the first Bell state as an example. We can write the equations which have to be satisfied for the state to be a tensor product.

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \otimes \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \tag{3.26}$$

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha\gamma \\ \alpha\delta \\ \beta\gamma \\ \beta\delta \end{pmatrix} \tag{3.27}$$

$$\tag{3.28}$$

However, these equations cannot be solved. The second equation states that

$$\alpha\delta = 0 \tag{3.29}$$

so that either $\alpha = 0$ or $\delta = 0$. If $\alpha = 0$, then the first equation cannot be satisfied, but if $\delta = 0$ then the last equation cannot be satisfied. A similar conclusion is reached looking at the third equation.

Next, lets write the first Bell state in the *sign basis*, meaning the basis that is rotated 45° from the standard basis,

$$\hat{\mathbf{e}}_+ = \frac{1}{\sqrt{2}}\left(\hat{\mathbf{e}}_0 + \hat{\mathbf{e}}_1\right), \tag{3.30}$$

$$\hat{\mathbf{e}}_- = \frac{1}{\sqrt{2}}\left(\hat{\mathbf{e}}_0 - \hat{\mathbf{e}}_1\right), \tag{3.31}$$

or in reverse

$$\hat{\mathbf{e}}_0 = \frac{1}{\sqrt{2}}\left(\hat{\mathbf{e}}_+ + \hat{\mathbf{e}}_-\right), \tag{3.32}$$

$$\hat{\mathbf{e}}_1 = \frac{1}{\sqrt{2}}\left(\hat{\mathbf{e}}_+ - \hat{\mathbf{e}}_-\right). \tag{3.33}$$

Now we can expand our Bell state in this new basis

$$B_0 = \frac{1}{\sqrt{2}}\left(\hat{\mathbf{e}}_{00} + \hat{\mathbf{e}}_{11}\right), \tag{3.34}$$

$$= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}\left(\hat{\mathbf{e}}_+ + \hat{\mathbf{e}}_-\right) \otimes \frac{1}{\sqrt{2}}\left(\hat{\mathbf{e}}_+ + \hat{\mathbf{e}}_-\right) + \hat{\mathbf{e}}_{11}\right), \tag{3.35}$$

$$= \frac{1}{2\sqrt{2}}\left(\hat{\mathbf{e}}_{++} + \hat{\mathbf{e}}_{+-} + \hat{\mathbf{e}}_{-+} + \hat{\mathbf{e}}_{--} + \hat{\mathbf{e}}_{11}\right), \tag{3.36}$$

$$= \frac{1}{2\sqrt{2}}\left(\hat{\mathbf{e}}_{++} + \hat{\mathbf{e}}_{+-} + \hat{\mathbf{e}}_{-+} + \hat{\mathbf{e}}_{--} + \hat{\mathbf{e}}_{++} - \hat{\mathbf{e}}_{+-} - \hat{\mathbf{e}}_{-+} + \hat{\mathbf{e}}_{--}\right), \tag{3.37}$$

$$= \frac{1}{\sqrt{2}}\left(\hat{\mathbf{e}}_{++} + \hat{\mathbf{e}}_{--}\right). \tag{3.38}$$

Thus we seem the same entanglement in this new basis. In fact, our state will be entangled in any basis.

3.5 Problems

Problem III.1 QALA 2.1

Problem III.2 QALA 2.3

Problem III.3 QALA 2.4

Problem III.4 QALA 2.5

Problem III.5 QALA 2.6

Problem III.6 QALA 2.7

Problem III.7 QALA 2.8

Problem III.8 QALA 4.2

Problem III.9 QALA 4.3

Problem III.10 QALA 4.4

Problem III.11 QALA 4.5

Problem III.12 QALA 4.6

Problem III.13 QALA 4.8

Problem III.14 QALA 4.9

Problem III.15 QALA 4.10

Problem III.16 QALA 4.12

Problem III.17 QALA 4.13

Problem III.18 QALA 4.14

Problem III.19 QALA 6.1

Problem III.20 QALA 6.2

Problem III.21 QALA 6.4

Problem III.22 QALA 6.5

Problem III.23 QALA 6.6

Problem III.24 QALA 6.7

Problem III.25 QALA 6.8

Problem III.26 QALA 6.9

Problem III.27 QALA 6.10

References

Hardy, Lucien (2001). "Quantum theory from five reasonable axioms". In: eprint: quant-ph/0101012.

Mallesh, KS, Subhash Chaturvedi, R Simon, and N Mukunda (2012). "States of physical systems in classical and quantum mechanics". In: *Resonance* 17.1, pp. 53–75.

Lipton, Richard J and Kenneth W Regan (2014). "Quantum Algorithms via Linear Algebra: A Primer". In: p. 206.

Chapter 4

Quantum Algorithms

The bag of tricks for quantum computing, I think, arises mainly, not from quantum properties, but rather from the necessity of reversibility. This is the origin, for instance, of the famous No-Cloning Theorem of quantum mechanics. This says that there does not exist a universal unitary transformation which produce an exact copy an unknown quantum state. To be specific, let us define a state $\mathbf{a}$, and ask for a transformation U_C such that

$$U_C(\mathbf{a} \otimes \hat{\mathbf{e}}_0) = e^{i\alpha(\mathbf{a},\hat{\mathbf{e}}_0)}(\mathbf{a} \otimes \mathbf{a})$$

Now suppose that we cloned two states a and b, and looked at their inner product

$$\begin{aligned}
(\mathbf{b} \otimes \hat{\mathbf{e}}_0)^\dagger(\mathbf{a} \otimes \hat{\mathbf{e}}_0) &= (\mathbf{b} \otimes e_0)^\dagger U_C^\dagger U_C(\mathbf{a} \otimes e_0) \\
(\mathbf{b}^\dagger\mathbf{a})(\hat{\mathbf{e}}_0^\dagger\hat{\mathbf{e}}_0) &= (U_C(\mathbf{b} \otimes e_0))^\dagger(U_C(\mathbf{a} \otimes e_0)) \\
(\mathbf{b}^\dagger\mathbf{a}) &= e^{-i\alpha(\mathbf{b},e_0)}(\mathbf{b} \otimes \mathbf{b})^\dagger e^{i\alpha(\mathbf{a},e_0)}(\mathbf{a} \otimes \mathbf{a}) \\
|\mathbf{b}^\dagger\mathbf{a}| &= |\mathbf{b}^\dagger\mathbf{a}|^2
\end{aligned}$$

This implies that either $\mathbf{b}^\dagger\mathbf{a} = 0$ or $\mathbf{b}^\dagger\mathbf{a} = 1$. Hence by the Cauchy-Schwarz Inequality the states are either parallel or orthogonal. This cannot be the case for two arbitrary states, and therefore, a single universal U_C cannot clone a general quantum state. Notice that we could design a U to copy a given quantum state, but the transformation would depend on the state it was copying.

We can, however, design a transform to clone only the basis vectors. If we feed $\hat{\mathbf{e}}_k \otimes \hat{\mathbf{e}}_0$ to **CNOT**, we get back $\hat{\mathbf{e}}_k \otimes \hat{\mathbf{e}}_k$. If we apply **CNOT** to successive pairs of qubits, we can create this state for $2n$ qubits. Lets look at the effect for a 2-qubit system. Suppose I apply **CNOT** to $\mathbf{a} \otimes \hat{\mathbf{e}}_0$,

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} a_0 \\ 0 \\ a_1 \\ 0 \end{pmatrix} = \begin{pmatrix} a_0 \\ 0 \\ 0 \\ a_1 \end{pmatrix}$$

so that $b_{ii} = a_i$. This means that for any basis vector, $U_C(e_k \otimes e_0) = e_k \otimes e_k$. However, lets apply this to the state $\frac{1}{\sqrt{2}}(e_0 + e_1) \otimes e_0$,

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

but that is not the cloned state

$$\frac{1}{\sqrt{2}}(e_0 + e_1) \otimes \frac{1}{\sqrt{2}}(e_0 + e_1) = \frac{1}{2} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

We can also do the computation symbolically

$$\begin{aligned} U_C \left(\frac{1}{\sqrt{2}}(e_0 + e_1) \otimes e_0 \right) &= \frac{1}{\sqrt{2}} U_C \left(e_0 \otimes e_0 + e_1 \otimes e_0 \right) \\ &= \frac{1}{\sqrt{2}} \left(e_0 \otimes e_0 + e_1 \otimes e_1 \right) \end{aligned}$$

whereas

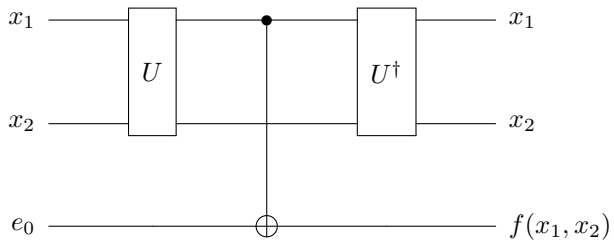
$$\frac{1}{\sqrt{2}}(e_0 + e_1) \otimes \frac{1}{\sqrt{2}}(e_0 + e_1) = \frac{1}{2}(e_0 \otimes e_0 + e_0 \otimes e_1 + e_1 \otimes e_0 + e_1 \otimes e_1).$$

This trick of copying the basis can be used to extract a subset of qubits in a reversible way. Suppose we want to select out some qubits from among the output of an operation U. This is not reversible, since we are discarding the other qubits. We want somehow to remove the information in the other qubits before we proceed. Suppose that we act with $U \otimes I$ on the initial state $a \otimes e_{0^m}$. Then pick the m qubits we want from that output using $\mathbf{C}_m$, where the **CNOT** controls are on our chosen qubits, and the targets are on the m ancillary qubits we are using to make things reversible. Then act with $U^\dagger \otimes I$ to return the input qubits to their original state. This is called the Copy-Uncompute trick, and can be expressed in our function notation as

$$\left(U^\dagger \otimes I \right) \mathbf{CNOT}_m \left(U \otimes I \right). \tag{4.1}$$

Let's try a simple example to see if we can predict the action using our tools. I want to act with a two qubits Boolean operator U, but only retain the result in the first qubit, say $f(x_1, x_2)$, whereas the second qubit was some $g(x_1, x_2)$.

The quantum circuit for this setup would be



In the function notation, this would be

$$\begin{aligned}
&\left(U^\dagger \otimes I\right) \mathbf{CNOT}_{1,3} \left(U \otimes I\right) \left(x_1 \otimes x_2 \otimes \hat{\mathbf{e}}_0\right) \\
=&\left(U^\dagger \otimes I\right) \mathbf{CNOT}_{1,3} \left(\hat{\mathbf{e}}_{f(x_1,x_2)} \otimes \hat{\mathbf{e}}_{g(x_1,x_2)} \otimes \hat{\mathbf{e}}_0\right) \\
=&\left(U^\dagger \otimes I\right) \left(\hat{\mathbf{e}}_{f(x_1,x_2)} \otimes \hat{\mathbf{e}}_{g(x_1,x_2)} \otimes \hat{\mathbf{e}}_{f(x_1,x_2)}\right) \\
=&x_1 \otimes x_2 \otimes \hat{\mathbf{e}}_{f(x_1,x_2)}
\end{aligned}$$

Notice that this works because the Boolean operator output is in a definite basis state. This is really just the same trick as making a reversible F for our non-invertible Boolean function, but done more explicitly with gates.

By linearity of the tensor product, scaling one part is equivalent to scaling the other parts

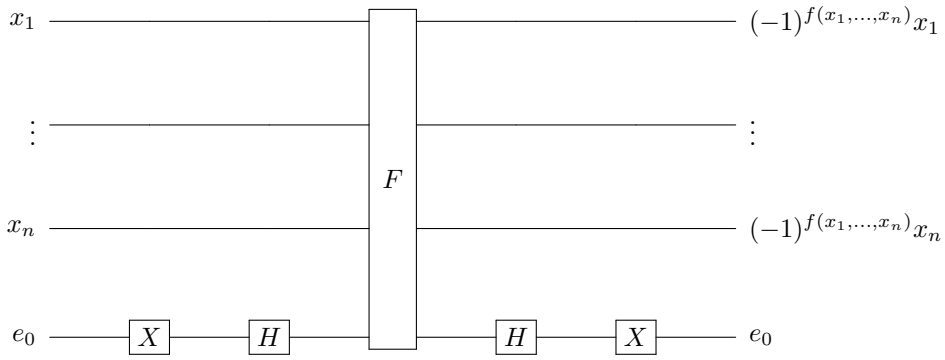
$$\alpha(\mathbf{v} \otimes \mathbf{w}) = \alpha\mathbf{v} \otimes \mathbf{w} = \mathbf{v} \otimes \alpha\mathbf{w}$$

We can show this explicitly by calculating the Kronecker product of two vectors,

$$\begin{aligned}
\alpha \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) &= \alpha \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ \alpha \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} \alpha \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ \alpha \end{pmatrix}
\end{aligned}$$

This is a strange property since it means we may move multipliers among the factors in a tensor product. Thus the vector $\mathbf{a} \otimes \mathbf{b}$ does not uniquely correspond to the tensor product of $\mathbf{a}$ and $\mathbf{b}$, but to all pairs of vectors $\alpha\mathbf{a}$ and $\alpha^{-1}\mathbf{b}$.

We can use this strangeness to get the effect of altering one output qubit, when it looks like we are altering another. Let F be the unitary operator created from a Boolean function $f(x_1, \ldots, x_n)$, and consider the circuit



The unitary operator F corresponding to the Boolean function f is linear, so that its action on mixed states is given by the linear combination of the action on pure states. Moreover, it is a permutation, mapping the input state labeled by xz to the output state labeled by $x(z \oplus f(x))$. In order to calculate the effect of the entire circuit, we first notice that

$$(I \otimes H)(I \otimes X)(x \otimes \hat{\mathbf{e}}_0) = x \otimes \frac{1}{\sqrt{2}}(e_0 - e_1) = \mathbf{x} \otimes \mathbf{d}$$

$$\begin{aligned} F(x,d) &= F(\mathbf{x} \otimes \mathbf{d}) \\ &= \frac{1}{\sqrt{2}}\left(F(\mathbf{x} \otimes \hat{\mathbf{e}}_0) - F(\mathbf{x} \otimes \hat{\mathbf{e}}_1)\right) \\ &= \frac{1}{\sqrt{2}}\left(F(x,0) - F(x,1)\right) \\ &= \frac{1}{\sqrt{2}}\left(\left(\mathbf{x} \otimes e_{0 \oplus f(x)}\right) - \left(\mathbf{x} \otimes e_{1 \oplus f(x)}\right)\right) \\ &= \frac{1}{\sqrt{2}}\left(\mathbf{x} \otimes \left(e_{0 \oplus f(x)} - e_{1 \oplus f(x)}\right)\right) \end{aligned}$$

Now we could look at the two cases, $f(x) = 0$ and $f(x) = 1$, so that

$$e_{0 \oplus f(x)} - e_{1 \oplus f(x)} = \begin{cases} e_0 - e_1 & f(x) = 0 \\ e_1 - e_0 & f(x) = 1 \end{cases}.$$

Looking at this, we can see a clever way to unify the classes, using a power of

minus one,

$$\begin{aligned} F(x,d) &= \frac{1}{\sqrt{2}}\left(\mathbf{x}\otimes(-1)^{f(x)}\left(e_0-e_1\right)\right) \\ &= (-1)^{f(x)}\left(\mathbf{x}\otimes\frac{1}{\sqrt{2}}\left(e_0-e_1\right)\right) \\ &= \left((-1)^{f(x)}\mathbf{x}\otimes\mathbf{d}\right) \end{aligned}$$

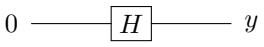
We have used the QALA notation $F(x,y)$ for $F(e_x\otimes e_y)$ in order to illustrate the book's notation in our format. Now if we finish off with the Hadamard and NOT gates (which is just the uncompute trick), we are left with the input vector $\mathbf{x}$ scaled by an x-dependent sign. Later we will see that this can be seen as a reflector, known as the *Grover Oracle*. This calculation shows that the Grover Oracle is feasible, since we can start with $\mathbf{j}$, and then act with this circuit to multiply all entries in the true set S with negative one.

4.1 Simple Examples

We would like to look at very simple circuits that often appear as subsystems for full algorithms. For each, we will analyze the quantum circuit using both linear algebra and Boolean algebra. We call the later analysis a *functional* approach since Boolean functions play a central role.

4.1.1 Create superposition

Quantum Circuit:



Linear Algebraic:

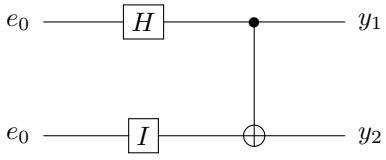
$$\frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} \tag{4.2}$$

Functional:

$$\mathbf{y} = \mathbf{H}e_0 = \frac{1}{\sqrt{2}}\left(e_0+e_1\right) \tag{4.3}$$

4.1.2 Create entanglement

Quantum Circuit:



Linear Algebraic:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \left(\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right) \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right) \tag{4.4}$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right) \tag{4.5}$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \tag{4.6}$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{4.7}$$

Functional:

$$\mathbf{y} = \mathbf{CNOT} \left(\mathbf{H} \otimes I \right) \left(\hat{\mathbf{e}}_0 \otimes \hat{\mathbf{e}}_0 \right) \tag{4.8}$$

$$= \mathbf{CNOT} \left(\mathbf{H} \hat{\mathbf{e}}_0 \otimes \hat{\mathbf{e}}_0 \right) \tag{4.9}$$

$$= \frac{1}{\sqrt{2}} \mathbf{CNOT} \left(\hat{\mathbf{e}}_0 \otimes \hat{\mathbf{e}}_0 + \hat{\mathbf{e}}_1 \otimes \hat{\mathbf{e}}_0 \right) \tag{4.10}$$

$$= \frac{1}{\sqrt{2}} \mathbf{CNOT} \left(\hat{\mathbf{e}}_{00} + \hat{\mathbf{e}}_{10} \right) \tag{4.11}$$

$$= \frac{1}{\sqrt{2}} \left(\hat{\mathbf{e}}_{00} + \hat{\mathbf{e}}_{11} \right) \tag{4.12}$$

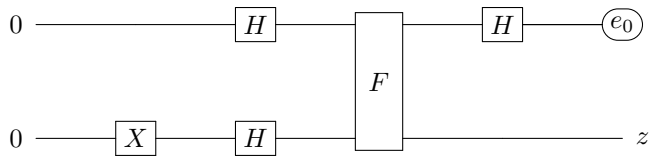
4.2 Deutsch's Algorithm

We would like to distinguish what is possible classically from quantum mechanically, but it has been hard to define exactly what kinds of computations are possible and fast on a quantum computer. Classically, the theory started with Gödel, Post, Church and recursively enumerable sets. It turned out that these sets were described by exactly the kinds of functions that people wanted to compute, and that this computation could be completely described by a Turing Machine (TM). Since we can build circuits out of logic gates for these computations, they are also equivalent to evaluating Boolean functions. We have seen that we can reversibly compute Boolean functions with little overhead, and that we can build quantum gates that mimic the classical gates for Boolean functions (with little overhead). Thus we can do any TM computation on a quantum computer. The difference is that we cannot read out the answer, but only make

a quantum measurement. If we model a quantum measurement as the action of a linear functional on the quantum state, we know that all such functionals can be realized as integrals by the Reisz-Markov-Kakutani Theorem. This is exactly what is happening in Deutsch's Algorithm and the Deutsch-Josza Algorithm.

We would like to determine whether a given unary Boolean function is constant. Since there are only four unary functions, we will distinguish between the constant functions (always true and always false) and the non-constant functions (identity and negation). Classically, you must measure the output for both possible inputs, since the outcome of one measurement will always be consistent with both a constant and non-constant function. However, Deutsch (Deutsch and Jozsa 1992) showed that a quantum computer could accomplish this in a single measurement.

The Deutsch algorithm, actually formulated in this final form by Deutsch and Jozsa, can be expressed by the following quantum circuit,



in which F is unitary operator corresponding to our unary Boolean function, and we introduce a new element on the first qubit. We will measure the probability that the first output qubit is $\hat{\mathbf{e}}_0$. In our analysis, this means we will sum over the probability of getting $\hat{\mathbf{e}}_0 \otimes z$ for any z.

First we will analyze the circuit using the linear algebraic representation. We have explictly formed matrices for the operators $I \otimes X$, $H \otimes H$, and $H \otimes I$.

$$\frac{1}{\sqrt{2}}\begin{pmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{pmatrix} U_F \frac{1}{2}\begin{pmatrix}1&1&1&1\\1&-1&1&-1\\1&1&-1&-1\\1&-1&-1&1\end{pmatrix}\begin{pmatrix}0&1&0&0\\1&0&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}1\\0\\0\\0\end{pmatrix}$$
$$=\frac{1}{\sqrt{2}}\begin{pmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{pmatrix} U_F \frac{1}{2}\begin{pmatrix}1&1&1&1\\1&-1&1&-1\\1&1&-1&-1\\1&-1&-1&1\end{pmatrix}\begin{pmatrix}0\\1\\0\\0\end{pmatrix}$$
$$=\frac{1}{2\sqrt{2}}\begin{pmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{pmatrix} U_F \begin{pmatrix}1\\-1\\1\\-1\end{pmatrix}$$

Now if f is the identity, then $F(x,z) = (x, x \oplus z)$ and

$$\begin{aligned}
&\frac{1}{2\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} U_F \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix} \\
=&\frac{1}{2\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix} \\
=&\frac{1}{2\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} \\
=&\frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 0 \\ 1 \\ -1 \end{pmatrix}.
\end{aligned}$$

Here, a measurement when the first qubit is 0 is just the projection on $\hat{\mathbf{e}}_0 \otimes z$,

$$\frac{1}{\sqrt{2}} \begin{pmatrix} z_0 & z_1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ -1 \end{pmatrix} = 0.$$

Similarly for negation, $F(x,z) = (x, (\neg x) \oplus z)$, we get

$$\begin{aligned}
&\frac{1}{2\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix} \\
=&\frac{1}{2\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} -1 \\ 1 \\ 1 \\ -1 \end{pmatrix} \\
=&\frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 0 \\ -1 \\ 1 \end{pmatrix}
\end{aligned}$$

with the same result. However for f the always true function, $F(x,z) = (x, \neg z)$,

we get

$$\begin{aligned}
&\frac{1}{2\sqrt{2}}\begin{pmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{pmatrix}\begin{pmatrix}0&1&0&0\\1&0&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}1\\-1\\1\\-1\end{pmatrix}\\
=&\frac{1}{2\sqrt{2}}\begin{pmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{pmatrix}\begin{pmatrix}-1\\1\\-1\\1\end{pmatrix}\\
=&\frac{1}{\sqrt{2}}\begin{pmatrix}-1\\1\\0\\0\end{pmatrix}
\end{aligned}$$

and a measurement when the first qubit is 0 will yield either state for the second qubit with probability 1/2, since

$$\begin{aligned}
\left|\frac{1}{\sqrt{2}}\begin{pmatrix}1&0&0&0\end{pmatrix}\begin{pmatrix}-1\\1\\0\\0\end{pmatrix}\right|^2&=\frac{1}{2},\\
\left|\frac{1}{\sqrt{2}}\begin{pmatrix}0&1&0&0\end{pmatrix}\begin{pmatrix}-1\\1\\0\\0\end{pmatrix}\right|^2&=\frac{1}{2}.
\end{aligned}$$

Similarly for the always false function,

$$\frac{1}{2\sqrt{2}}\begin{pmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}\begin{pmatrix}1\\-1\\1\\-1\end{pmatrix} \tag{4.13}$$

$$=\frac{1}{2\sqrt{2}}\begin{pmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{pmatrix}\begin{pmatrix}1\\-1\\1\\-1\end{pmatrix} \tag{4.14}$$

$$=\frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\\0\\0\end{pmatrix} \tag{4.15}$$

If we use the functional representation, we can get an even more compact

representation.

$$(\mathbf{H} \otimes I)\, F\, (\mathbf{H} \otimes \mathbf{H})\, (\hat{\mathbf{e}}_0 \otimes \hat{\mathbf{e}}_1) \tag{4.16}$$

$$= (\mathbf{H} \otimes I)\, F \frac{1}{2}\, ((\hat{\mathbf{e}}_0 + \hat{\mathbf{e}}_1) \otimes (\hat{\mathbf{e}}_0 - \hat{\mathbf{e}}_1)) \tag{4.17}$$

$$= \frac{1}{2}\, (\mathbf{H} \otimes I)\, F\, (\hat{\mathbf{e}}_{00} - \hat{\mathbf{e}}_{01} + \hat{\mathbf{e}}_{10} - \hat{\mathbf{e}}_{11}) \tag{4.18}$$

$$= \frac{1}{2}\, (\mathbf{H} \otimes I)\, \left(\hat{\mathbf{e}}_{0f(0)} - \hat{\mathbf{e}}_{0\neg f(0)} + \hat{\mathbf{e}}_{1f(1)} - \hat{\mathbf{e}}_{1\neg f(1)}\right) \tag{4.19}$$

$$= \frac{1}{2\sqrt{2}} (\hat{\mathbf{e}}_{0f(0)} + \hat{\mathbf{e}}_{1f(0)} - \hat{\mathbf{e}}_{0\neg f(0)} - \hat{\mathbf{e}}_{1\neg f(0)} + \hat{\mathbf{e}}_{0f(1)} - \hat{\mathbf{e}}_{1f(1)} - \hat{\mathbf{e}}_{0\neg f(1)} + \hat{\mathbf{e}}_{1\neg f(1)}) \tag{4.20}$$

If f is constant, $f(0) = f(1) = y$, we have

$$\frac{1}{2\sqrt{2}}\, (\hat{\mathbf{e}}_{0y} + \hat{\mathbf{e}}_{1y} - \hat{\mathbf{e}}_{0\neg y} - \hat{\mathbf{e}}_{1\neg y} + \hat{\mathbf{e}}_{0y} - \hat{\mathbf{e}}_{1y} - \hat{\mathbf{e}}_{0\neg y} + \hat{\mathbf{e}}_{1\neg y}) \tag{4.21}$$

$$= \frac{1}{\sqrt{2}}\, (\hat{\mathbf{e}}_{0y} - \hat{\mathbf{e}}_{0\neg y}) \tag{4.22}$$

whereas if f is not constant, $f(0) = y$ and $f(1) = \neg y$, we have

$$\frac{1}{2\sqrt{2}}\, (\hat{\mathbf{e}}_{0y} + \hat{\mathbf{e}}_{1y} - \hat{\mathbf{e}}_{0\neg y} - \hat{\mathbf{e}}_{1\neg y} + \hat{\mathbf{e}}_{0\neg y} - \hat{\mathbf{e}}_{1\neg y} - \hat{\mathbf{e}}_{0y} + \hat{\mathbf{e}}_{1y}) \tag{4.23}$$

$$= \frac{1}{\sqrt{2}}\, (\hat{\mathbf{e}}_{1y} - \hat{\mathbf{e}}_{1\neg y}) \tag{4.24}$$

We can simplify this derivation by using the expression for the action of the Hadamard operator from QALA. Remember that for systems of qubits, we can write our linear algebraic indices as Boolean strings. Thus if we have $\mathbf{b} = H_4\mathbf{a}$, we can index $\mathbf{a}$ by the bit string $\mathbf{z}$ and $\mathbf{b}$ by $\mathbf{x}$, so that

$$\mathbf{b}_{\mathbf{x}} = \frac{1}{2} \sum_{\mathbf{z}} -1^{\mathbf{x}\cdot\mathbf{z}} \mathbf{a}_{\mathbf{z}}.$$

If our starting state is $\hat{\mathbf{e}}_{01}$, then we have

$$\begin{aligned} \mathbf{b}_{\mathbf{x}} &= \frac{1}{2}(-1)^{\mathbf{x}\cdot 01} \\ &= \frac{1}{2}(-1)^{x\cdot 0 \oplus y\cdot 1} \\ &= \frac{1}{2}(-1)^{y} \end{aligned}$$

where we have used the properties of powers of minus one which you proved in the homework. Now we use the definition of our reversible Boolean function F,

$$F = \sum_{xz} (\hat{\mathbf{e}}_x \otimes \hat{\mathbf{e}}_z)(\hat{\mathbf{e}}_x^\dagger \otimes \hat{\mathbf{e}}_{z\oplus f(x)}^\dagger)$$

we have $\mathbf{c} = F\mathbf{b}$,

$$\begin{aligned}\mathbf{c}_{xy} &= \sum_{wz} F_{xy,wz}\mathbf{b}_{wz} \\ &= \mathbf{b}_{x(y\oplus f(x))} \\ &= \frac{1}{2}(-1)^{y\oplus f(x)}.\end{aligned}$$

Then we finally apply a Hadmard gate only on the first quibit, $\mathbf{d} = (H_2 \otimes I)\mathbf{c}$,

$$\begin{aligned}\mathbf{d}_{xy} &= \frac{1}{\sqrt{2}}\sum_{wz}(-1)^{xw}\delta_{yz}\mathbf{c}_{wz} \\ &= \frac{1}{2\sqrt{2}}\sum_{wz}(-1)^{xw}\delta_{yz}(-1)^{z\oplus f(w)} \\ &= \frac{1}{2\sqrt{2}}\sum_{w}(-1)^{xw}(-1)^{y\oplus f(w)}.\end{aligned}$$

Now we are again reduced to plugging in values

$$\begin{aligned}\mathbf{d}_{xy} &= \frac{1}{2\sqrt{2}}\left((-1)^{y\oplus f(0)} + (-1)^{x}(-1)^{y\oplus f(1)}\right) \\ &= \frac{1}{2\sqrt{2}}(-1)^{y}\left((-1)^{f(0)} + (-1)^{x\oplus f(1)}\right).\end{aligned}$$

Now we see that the amplitude for state $\mathbf{d}_{0y}$ is given by

$$\frac{1}{8}\left|-1^{f(0)} + -1^{f(1)}\right|^2$$

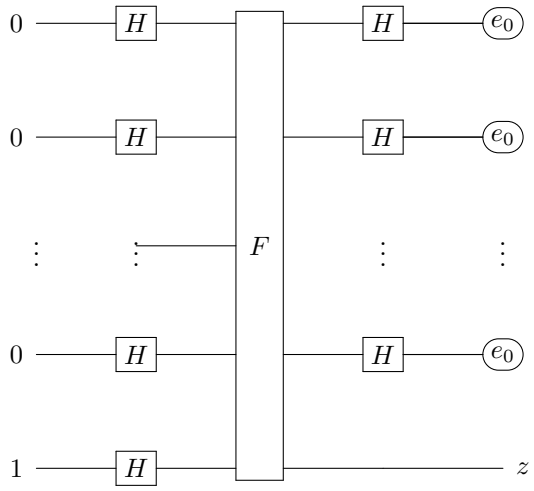
so that if f is constant, we have amplitude one half for either state y, whereas if the function changes then the amplitude is 0.

4.3 Deutsch-Jozsa Algorithm

The Deutsch-Jozsa algorithm is just the Deutsch repeated for n qubits. This time we are looking to discriminate between constant functions and *balanced functions*, which are functions having an equal number of true and false outcomes. If I just sample classically, I would need to look at one more than half the bits in order to assure myself that the function is not constant. However, the quantum algorithm can distinguish between them with a single sample.

The circuit for this algorithm is given below. The strategy is the same as before, namely to act with a Hadamard gate on all inputs, then the Boolean function, and then with a final Hadamard gate on all outputs except z. We

again measure the probability for $0z$, which will vanish for balanced functions.



This example shows the power of our functional notation. Treating an arbitrary number of qubits in the linear algebraic notation is quite cumbersome.

$$(\mathbf{H}\cdots\otimes\mathbf{H}\otimes I)\,F\,(\mathbf{H}\otimes\cdots\otimes\mathbf{H})\,(\hat{\mathbf{e}}_0\otimes\cdots\otimes\hat{\mathbf{e}}_0\otimes\hat{\mathbf{e}}_1) \tag{4.25}$$

$$=\frac{1}{\sqrt{2N}}\,(\mathbf{H}\cdots\otimes\mathbf{H}\otimes I)\,F\sum_{(\mathbf{x},z)=\mathbf{0}}^{2N-1}e_{\mathbf{x}z}\sum_{(\mathbf{x}',z')=\mathbf{0}}^{2N-1}(-1)^{(\mathbf{x},z)\cdot(\mathbf{x}',z')}\delta_{\mathbf{x}'\mathbf{0}}\delta_{z'1} \tag{4.26}$$

$$=\frac{1}{\sqrt{2N}}\,(\mathbf{H}\cdots\otimes\mathbf{H}\otimes I)\,F\sum_{(\mathbf{x},z)=\mathbf{0}}^{2N-1}e_{\mathbf{x}z}(-1)^{(\mathbf{x},z)\cdot(\mathbf{0},1)} \tag{4.27}$$

$$=\frac{1}{\sqrt{2N}}\,(\mathbf{H}\cdots\otimes\mathbf{H}\otimes I)\,F\sum_{(\mathbf{x},z)=\mathbf{0}}^{2N-1}e_{\mathbf{x}z}(-1)^{\mathbf{x}\cdot\mathbf{0}}(-1)^{z\cdot 1} \tag{4.28}$$

$$=\frac{1}{\sqrt{2N}}\,(\mathbf{H}\cdots\otimes\mathbf{H}\otimes I)\,F\sum_{(\mathbf{x},z)=\mathbf{0}}^{2N-1}e_{\mathbf{x}z}(-1)^{z} \tag{4.29}$$

$$=\frac{1}{\sqrt{2N}}\,(\mathbf{H}\cdots\otimes\mathbf{H}\otimes I)\sum_{(\mathbf{x},z)=\mathbf{0}}^{2N-1}e_{\mathbf{x}z}(-1)^{z\oplus f(\mathbf{x})} \tag{4.30}$$

$$=\frac{1}{\sqrt{2N}}\frac{1}{\sqrt{N}}\left(\sum_{\mathbf{x}'=\mathbf{0}}^{N-1}\sum_{\mathbf{x}=\mathbf{0}}^{N-1}(-1)^{\mathbf{x}\cdot\mathbf{x}'}e_{\mathbf{x}}\right)\otimes\sum_{z}e_z(-1)^{z\oplus f(\mathbf{x}')} \tag{4.31}$$

$$=\frac{1}{\sqrt{2}N}\sum_{(\mathbf{x},z)=\mathbf{0}}^{2N-1}\sum_{\mathbf{x}'=\mathbf{0}}^{N-1}(-1)^{\mathbf{x}\cdot\mathbf{x}'}(-1)^{z\oplus f(\mathbf{x}')}e_{\mathbf{x}z} \tag{4.32}$$

Thus, the probability of measuring a state (x, z) is given by

$$P(x,z) = \frac{1}{2N^2}\left|\sum_{\mathbf{x}'=\mathbf{0}}^{N-1}(-1)^{\mathbf{x}\cdot\mathbf{x}'}(-1)^{z\oplus f(\mathbf{x}')}\right|^2, \tag{4.33}$$

$$= \frac{1}{2N^2}\left|(-1)^z\sum_{\mathbf{x}'=0}^{N-1}(-1)^{\mathbf{x}\cdot\mathbf{x}'}(-1)^{f(\mathbf{x}')}\right|^2, \tag{4.34}$$

$$= \frac{1}{2N^2}\left|\sum_{\mathbf{x}'=0}^{N-1}(-1)^{\mathbf{x}\cdot\mathbf{x}'}(-1)^{f(\mathbf{x}')}\right|^2. \tag{4.35}$$

If we ask for $P(\mathbf{0}, z)$, we get

$$P(\mathbf{0},z) = \frac{1}{2N^2}\left|\sum_{\mathbf{x}'=\mathbf{0}}^{N-1}(-1)^{\mathbf{0}\cdot\mathbf{x}'}(-1)^{f(\mathbf{x}')}\right|^2, \tag{4.36}$$

$$= \frac{1}{2N^2}\left|\sum_{\mathbf{x}'=\mathbf{0}}^{N-1}(-1)^{f(\mathbf{x}')}\right|^2. \tag{4.37}$$

If f is constant, then we get

$$P(\mathbf{0},z) = \frac{1}{2N^2}\left|\sum_{x'=\mathbf{0}}^{N-1}\pm 1\right|^2, \tag{4.38}$$

$$= \frac{1}{2N^2}\left|\pm N\right|^2, \tag{4.39}$$

$$= \frac{1}{2}, \tag{4.40}$$

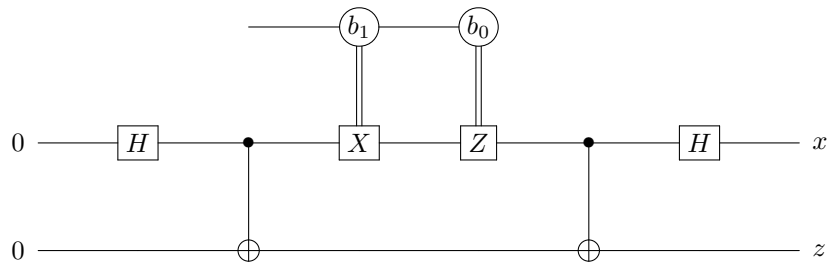
whereas, if f is balanced, then half the terms in the sum cancel the other half, and we get zero.

4.4 Superdense coding

There are several algorithms similar to the Deutsch-Josza algorithms, in that they begin with a superposition or entanglement, act with a Boolean function, and then uncompute the initial action. This pattern is followed in *superdense coding*, so named because we can send two classical bits using a single qubit gate.

We start by producing the Bell state, as shown in the circuit below. We then use two classical bits to decide which transformation to apply to the first qubit. If the classical bit is true, the connected gate is active. For example, if both bits are true, we apply $U = \mathbf{ZX}$ to the input. Finally, we uncompute the entanglement. Since the qubits were entangled when we acted on the first

qubit, information can be retrieved from both.



First we work through the case when the transformation U is the identity, meaning $b_0 = b_1 = 0$, so we expect non-zero amplitude only in the (00) state,

$$(\mathbf{H} \otimes I)\mathbf{CNOT}(\mathbf{U} \otimes I)\mathbf{CNOT}(\mathbf{H} \otimes I)\hat{\mathbf{e}}_{00} \tag{4.41}$$
$$= \frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)\mathbf{CNOT}\ \mathbf{CNOT}((\hat{\mathbf{e}}_0 + \hat{\mathbf{e}}_1) \otimes \hat{\mathbf{e}}_0) \tag{4.42}$$
$$= \frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)\mathbf{CNOT}\ \mathbf{CNOT}(\hat{\mathbf{e}}_{00} + \hat{\mathbf{e}}_{10}) \tag{4.43}$$
$$= \frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)\mathbf{CNOT}(\hat{\mathbf{e}}_{00} + \hat{\mathbf{e}}_{11}) \tag{4.44}$$
$$= \frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)(\hat{\mathbf{e}}_{00} + \hat{\mathbf{e}}_{10}) \tag{4.45}$$
$$= \frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)((\hat{\mathbf{e}}_0 + \hat{\mathbf{e}}_1) \otimes \hat{\mathbf{e}}_0) \tag{4.46}$$
$$= (\hat{\mathbf{e}}_0 \otimes \hat{\mathbf{e}}_0) \tag{4.47}$$
$$= \hat{\mathbf{e}}_{00} \tag{4.48}$$

Then $U = X$ which gives (01),

$$\frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)\mathbf{CNOT}(\mathbf{X} \otimes I)(\hat{\mathbf{e}}_{00} + \hat{\mathbf{e}}_{11}) \tag{4.49}$$
$$= \frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)\mathbf{CNOT}(\hat{\mathbf{e}}_{10} + \hat{\mathbf{e}}_{01}) \tag{4.50}$$
$$= \frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)(\hat{\mathbf{e}}_{11} + \hat{\mathbf{e}}_{01}) \tag{4.51}$$
$$= \frac{1}{\sqrt{2}}(\mathbf{H} \otimes I)((\hat{\mathbf{e}}_0 + \hat{\mathbf{e}}_1) \otimes \hat{\mathbf{e}}_1) \tag{4.52}$$
$$= (\hat{\mathbf{e}}_0 \otimes \hat{\mathbf{e}}_1) \tag{4.53}$$
$$= \hat{\mathbf{e}}_{01} \tag{4.54}$$

$U = Z$ gives (10),

$$\begin{aligned}
&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)\mathbf{CNOT}(\mathbf{Z}\otimes I)(\hat{\mathbf{e}}_{00}+\hat{\mathbf{e}}_{11}) && (4.55)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)\mathbf{CNOT}(\hat{\mathbf{e}}_{00}-\hat{\mathbf{e}}_{11}) && (4.56)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)(\hat{\mathbf{e}}_{00}-\hat{\mathbf{e}}_{10}) && (4.57)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)((\hat{\mathbf{e}}_{0}-\hat{\mathbf{e}}_{1})\otimes\hat{\mathbf{e}}_{0}) && (4.58)\\
=&(\hat{\mathbf{e}}_{1}\otimes\hat{\mathbf{e}}_{0}) && (4.59)\\
=&\hat{\mathbf{e}}_{10} && (4.60)
\end{aligned}$$

and $U = ZX$ gives (11),

$$\begin{aligned}
&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)\mathbf{CNOT}(\mathbf{ZX}\otimes I)(\hat{\mathbf{e}}_{00}+\hat{\mathbf{e}}_{11}) && (4.61)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)\mathbf{CNOT}(\hat{\mathbf{e}}_{01}-\hat{\mathbf{e}}_{10}) && (4.62)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)(\hat{\mathbf{e}}_{01}-\hat{\mathbf{e}}_{11}) && (4.63)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)((\hat{\mathbf{e}}_{0}-\hat{\mathbf{e}}_{1})\otimes\hat{\mathbf{e}}_{1}) && (4.64)\\
=&(\hat{\mathbf{e}}_{1}\otimes\hat{\mathbf{e}}_{1}) && (4.65)\\
=&\hat{\mathbf{e}}_{11} && (4.66)
\end{aligned}$$

In order to work this out in the general case, we need an expression for U parameterized by the values of the bits $(b_0 b_1)$ that we would like to send across the channel. Let us define the single qubit operator U as

$$U = \begin{pmatrix} (-1)^{b_0\cdot b_1}\neg b_0 & (-1)^{\neg b_0\cdot b_1} & b_0 \\ (-1)^{b_0\cdot b_1} & b_0 & (-1)^{\neg b_0\cdot b_1}\neg b_0 \end{pmatrix}$$

or equivalently

$$U = (-1)^{b_0\cdot b_1}\hat{\mathbf{e}}_{b_1}\hat{\mathbf{e}}_0^\dagger + (-1)^{b_0\cdot\neg b_1}\hat{\mathbf{e}}_{\neg b_1}\hat{\mathbf{e}}_1^\dagger \qquad (4.67)$$

Now we can write this out in the general case

$$
\begin{aligned}
&(\mathbf{H}\otimes I)\mathbf{CNOT}(\mathbf{U}\otimes I)\mathbf{CNOT}(\mathbf{H}\otimes I)\hat{\mathbf{e}}_{00} && (4.68)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)\mathbf{CNOT}(\mathbf{U}\otimes I)\mathbf{CNOT}((\hat{\mathbf{e}}_0+\hat{\mathbf{e}}_1)\otimes\hat{\mathbf{e}}_0) && (4.69)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)\mathbf{CNOT}(\mathbf{U}\otimes I)\mathbf{CNOT}(\hat{\mathbf{e}}_{00}+\hat{\mathbf{e}}_{10}) && (4.70)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)\mathbf{CNOT}(\mathbf{U}\otimes I)(\hat{\mathbf{e}}_{00}+\hat{\mathbf{e}}_{11}) && (4.71)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)\mathbf{CNOT}((-1)^{b_0\cdot b_1}\hat{\mathbf{e}}_{b_10}+(-1)^{b_0\cdot\neg b_1}\hat{\mathbf{e}}_{\neg b_1 1}) && (4.72)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)(\neg b_1\hat{\mathbf{e}}_{00}+b_1(-1)^{b_0}\hat{\mathbf{e}}_{11}+\neg b_1(-1)^{b_0}\hat{\mathbf{e}}_{10}+b_1\hat{\mathbf{e}}_{01}) && (4.73)\\
=&\frac{1}{\sqrt{2}}(\mathbf{H}\otimes I)((\neg b_1\hat{\mathbf{e}}_0+\neg b_1(-1)^{b_0}\hat{\mathbf{e}}_1)\otimes\hat{\mathbf{e}}_0+(b_1(-1)^{b_0}\hat{\mathbf{e}}_1+b_1\hat{\mathbf{e}}_0)\otimes\hat{\mathbf{e}}_1) && (4.74)\\
=&((\neg b_1(\hat{\mathbf{e}}_0+\hat{\mathbf{e}}_1)+\neg b_1(-1)^{b_0}(\hat{\mathbf{e}}_0-\hat{\mathbf{e}}_1))\otimes\hat{\mathbf{e}}_0\\
&+(b_1(-1)^{b_0}(\hat{\mathbf{e}}_0-\hat{\mathbf{e}}_1)+b_1(\hat{\mathbf{e}}_0+\hat{\mathbf{e}}_1))\otimes\hat{\mathbf{e}}_1) && (4.75)\\
=&\neg b_1\frac{1+(-1)^{b_0}}{2}\hat{\mathbf{e}}_{00}+\neg b_1\frac{1-(-1)^{b_0}}{2}\hat{\mathbf{e}}_{10}\\
&+b_1\frac{1+(-1)^{b_0}}{2}\hat{\mathbf{e}}_{01}+b_1\frac{1-(-1)^{b_0}}{2}\hat{\mathbf{e}}_{11} && (4.76)
\end{aligned}
$$

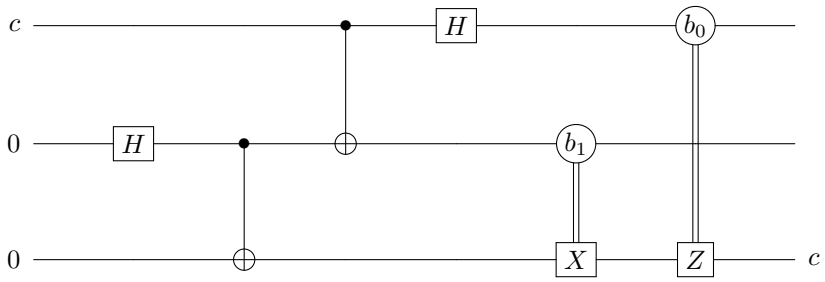
and check that we obtain the expected amplitudes.

$$
\begin{aligned}
(0,0)&\rightarrow\hat{\mathbf{e}}_{00}\\
(0,1)&\rightarrow\hat{\mathbf{e}}_{01}\\
(1,0)&\rightarrow\hat{\mathbf{e}}_{10}\\
(1,1)&\rightarrow\hat{\mathbf{e}}_{11}
\end{aligned}
$$

4.5 Quantum Teleportation

Quantum teleportation is the replication of an unknown quantum state, perhaps at a remote location. On the first qubit, the sender has an unknown state c. The second qubit, owned by the sender, is entangled with the third qubit, owned by the received. The sender measures the first two qubits after the familiar operations, and send those bits to the receiver. The receiver can then decide what transformation to apply in order to reconstruct the state c on the third qubit. Note that we have to destroy the state c at the sender (by measuring it) in order to reconstruct it at the receiver. Also, we use the same operator at the receiver as we used to do encoding in our Superdense coding circuit.

Quantum Circuit:



Let us define the qubit $\mathbf{c} = \alpha\hat{\mathbf{e}}_0 + \beta\hat{\mathbf{e}}_1$, and then write the functional expression for our circuit, evaluating the circuit up to the measurement step,

$$(\mathbf{H}\otimes I\otimes I)(\mathbf{CNOT}\otimes I)(I\otimes\mathbf{CNOT})(I\otimes\mathbf{H}\otimes I)(\mathbf{c}\otimes\hat{\mathbf{e}}_0\otimes\hat{\mathbf{e}}_0) \tag{4.77}$$

$$=(\mathbf{H}\otimes I\otimes I)(\mathbf{CNOT}\otimes I)(I\otimes\mathbf{CNOT})\frac{1}{\sqrt{2}}(\mathbf{c}\otimes(\hat{\mathbf{e}}_0+\hat{\mathbf{e}}_1)\otimes\hat{\mathbf{e}}_0) \tag{4.78}$$

$$=(\mathbf{H}\otimes I\otimes I)(\mathbf{CNOT}\otimes I)\frac{1}{\sqrt{2}}(\mathbf{c}\otimes\hat{\mathbf{e}}_{00}+\mathbf{c}\otimes\hat{\mathbf{e}}_{11}) \tag{4.79}$$

$$=(\mathbf{H}\otimes I\otimes I)(\mathbf{CNOT}\otimes I)\frac{1}{\sqrt{2}}(\alpha\hat{\mathbf{e}}_{000}+\beta\hat{\mathbf{e}}_{100}+\alpha\hat{\mathbf{e}}_{011}+\beta\hat{\mathbf{e}}_{111}) \tag{4.80}$$

$$=(\mathbf{H}\otimes I\otimes I)\frac{1}{\sqrt{2}}(\alpha\hat{\mathbf{e}}_{000}+\beta\hat{\mathbf{e}}_{110}+\alpha\hat{\mathbf{e}}_{011}+\beta\hat{\mathbf{e}}_{101}) \tag{4.81}$$

$$=\frac{1}{2}(\alpha\hat{\mathbf{e}}_{000}+\alpha\hat{\mathbf{e}}_{100}+\beta\hat{\mathbf{e}}_{010}-\beta\hat{\mathbf{e}}_{110}+\alpha\hat{\mathbf{e}}_{011}+\alpha\hat{\mathbf{e}}_{111}+\beta\hat{\mathbf{e}}_{001}-\beta\hat{\mathbf{e}}_{101}) \tag{4.82}$$

$$=\frac{1}{2}\left(\hat{\mathbf{e}}_{00}\otimes(\alpha\hat{\mathbf{e}}_0+\beta\hat{\mathbf{e}}_1)+\hat{\mathbf{e}}_{01}\otimes(\beta\hat{\mathbf{e}}_0+\alpha\hat{\mathbf{e}}_1)\right)$$

$$+\frac{1}{2}\left(\hat{\mathbf{e}}_{10}\otimes(\alpha\hat{\mathbf{e}}_0-\beta\hat{\mathbf{e}}_1)+\hat{\mathbf{e}}_{11}\otimes(-\beta\hat{\mathbf{e}}_0+\alpha\hat{\mathbf{e}}_1)\right) \tag{4.83}$$

Now if we measure the first two qubits, we select one of the states above. We could analyze this case-by-case, but instead let us write a parameterized state based on the bits $(b_0 b_1)$ we get from the measurement

$$\frac{1}{2}\left(\hat{\mathbf{e}}_{b_0 b_1}\otimes(\alpha\hat{\mathbf{e}}_{b_1}+(-1)^{b_0}\beta\hat{\mathbf{e}}_{\neg b_1})\right)$$

and we can act on the last qubit using the operator we defined in Eq. (4.67),

$$\left((-1)^{b_0\cdot b_1}\hat{\mathbf{e}}_{b_1}\hat{\mathbf{e}}_0^\dagger+(-1)^{b_0\cdot\neg b_1}\hat{\mathbf{e}}_{\neg b_1}\hat{\mathbf{e}}_1^\dagger\right)(\alpha\hat{\mathbf{e}}_{b_1}+(-1)^{b_0}\beta\hat{\mathbf{e}}_{\neg b_1}) \tag{4.84}$$

$$=\neg b_1((-1)^{b_0\cdot b_1}\alpha\hat{\mathbf{e}}_{b_1}+(-1)^{b_0\cdot\neg b_1}(-1)^{b_0}\beta\hat{\mathbf{e}}_{\neg b_1})$$

$$+\ \ b_1((-1)^{b_0\cdot b_1}(-1)^{b_0}\beta\hat{\mathbf{e}}_{b_1}+(-1)^{b_0\cdot\neg b_1}\alpha\hat{\mathbf{e}}_{\neg b_1}) \tag{4.85}$$

$$=\neg b_1(\alpha\hat{\mathbf{e}}_0+\beta\hat{\mathbf{e}}_1)+b_1(\beta\hat{\mathbf{e}}_1+\alpha\hat{\mathbf{e}}_0) \tag{4.86}$$

$$=(\neg b_1+b_1)(\alpha\hat{\mathbf{e}}_0+\beta\hat{\mathbf{e}}_1) \tag{4.87}$$

$$=\alpha\hat{\mathbf{e}}_0+\beta\hat{\mathbf{e}}_1 \tag{4.88}$$

$$=\mathbf{c} \tag{4.89}$$

and we have exactly recovered the original state of the first qubit, now in the third qubit.

4.6 Grover's Algorithm

There is a good description of Grover's Algorithm in QALA, but the algorithmic parts are somewhat spread out in the book. We will attempt to bring it all together in these notes. First, the point of the algorithm is to pick an item from an unordered N-item list. This item should satisfy some "hit" criterion, which we assume can be encoded in a Boolean function f. Thus, if I have some item in the list, I can feed it to f and determine if it is in our special set S of solutions. This also means that f must be feasible, but f^{-1} cannot be feasible, otherwise I could get at least some member of S by computing $f^{-1}(1)$. This kind of asymmetry should remind us of hash functions, and indeed we could try to find some number whose hash had a known value using Grover's algorithm. Note that S is refered to as the *characteristic set* of f.

Suppose there are a large number N of possible answers, and $k \ll N$ correct solutions. Classically, in the worst case, I would need to test $N-k$ items before I hit a solution. If I randomly guess, I could cut this down to $N/2-k$ guesses. However, we will see that I can use $\mathcal{O}(\sqrt{N})$ iterations of Grover's algorithm to have a good chance of finding a solution.

The key insight for this algorithm is linear algebraic at heart. We will create a vector space where each coordinate direction represents a possible guess, so that the space has dimension N. Suppose we define the characteristic vector $\hat{\mathbf{h}}$ for S, or *hit* vector,

$$\hat{\mathbf{h}}_i = \begin{cases} \frac{1}{\sqrt{k}} & i \in S \\ 0 & \text{otherwise} \end{cases}$$

where $k = |S|$, and the orthogonal *miss* vector $\hat{\mathbf{m}}$,

$$\hat{\mathbf{m}}_i = \begin{cases} 0 & i \in S \\ \frac{1}{\sqrt{N-k}} & \text{otherwise} \end{cases}.$$

Note that we can make the constant vector $\hat{\mathbf{j}}$ from these two

$$\hat{\mathbf{j}} = \frac{1}{\sqrt{N}} \left(\sqrt{k}\, \hat{\mathbf{h}} + \sqrt{N-k}\, \hat{\mathbf{m}} \right).$$

That means that $\hat{\mathbf{h}}$, $\hat{\mathbf{m}}$, and $\hat{\mathbf{j}}$ all lie in a two-dimensional subspace of our original N-dimensional space. If we imagine $\hat{\mathbf{m}}$ as the x-axis, and $\hat{\mathbf{h}}$ as the y-axis, then $\hat{\mathbf{j}}$ is in the positive quadrant between them. We can easily begin with $\hat{\mathbf{j}}$, so our aim is to rotate this vector into $\hat{\mathbf{h}}$, which when measured will assure us of a solution.

If we have coplanar vectors separated by an angle α, then a reflection about each one in sequence yields a rotation of angle 2α (the reader should prove this). We know the angle between $\hat{\mathbf{j}}$ and $\hat{\mathbf{m}}$,

$$\cos(\alpha) = \hat{\mathbf{j}}^\dagger \hat{\mathbf{m}} = \sqrt{\frac{N-k}{N}},$$

or we could write this as $\sin^2(\alpha) = k/N$, so That

$$\alpha = \sin^{-1}\left(\frac{k}{N}\right).$$

Since $k \ll N$, we see that $\hat{\mathbf{j}}$ is very close to $\hat{\mathbf{m}}$, and far from $\hat{\mathbf{h}}$. In fact, if we rotate by 2α for t_k times, where

$$\begin{aligned} 2\alpha t_k + \alpha &= \frac{\pi}{2} \\ t_k &= \frac{\pi}{4\alpha} - \frac{1}{2} \\ t_k &\cong \lfloor \frac{\pi}{4\alpha} \rfloor \end{aligned}$$

We can reflect about $\hat{\mathbf{j}}$ using the simple reflector definition $2\hat{\mathbf{j}}\hat{\mathbf{j}}^\dagger - I$ since we have an explicit representation of the vector. However, we do not know the miss vector $\hat{\mathbf{m}}$, since if we did then we would know $\hat{\mathbf{h}}$ and have already solved the problem. We will show, in a few steps, that this reflection can be feasibly computed by a quantum circuit. To start, we observe that any vector in our two-dimensional subspace is constant for directions in S, and for directions not in S. This is true because it is true for $\hat{\mathbf{h}}$ and $\hat{\mathbf{m}}$, and thus must be true for any linear combination. QALA calls this the *solution smoothness* property.

Second, we will look at the reflection of a vector with the solution smoothness property about $\hat{\mathbf{m}}$. So we take a vector $\mathbf{v}$ where every entry not in S has value e. Then we have

$$\hat{\mathbf{m}}^\dagger \mathbf{v} = \frac{e(N-k)}{\sqrt{N-k}} = e\sqrt{N-k},$$

which means that the projection of $\mathbf{v}$ on $\hat{\mathbf{m}}$ is

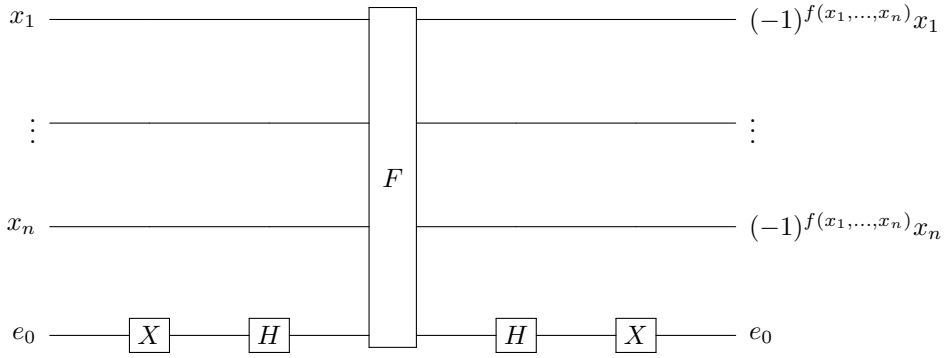
$$\left(\hat{\mathbf{m}}\hat{\mathbf{m}}^\dagger \mathbf{v}\right)_i = \begin{cases} 0 & i \in S \\ e & \text{otherwise} \end{cases}.$$

This means that a reflection about $\hat{\mathbf{m}}$ would give

$$\left(2\hat{\mathbf{m}}\hat{\mathbf{m}}^\dagger \mathbf{v} - \mathbf{v}\right)_i = \begin{cases} -v_i & i \in S \\ v_i & \text{otherwise} \end{cases},$$

so that we have negated all coefficients of directions in our solution set S. We can think of this as a large diagonal matrix, with -1 on the diagonal for each solution

state, and 1 for all other states. We will prove that this is feasible by explicitly constructing the quantum circuit, which is refered to as the *Grover Oracle*. We note here that even though they are simple, not all diagonal operators are feasible, since there are simply too many to be constructable by a polynomial number of gates. At the start of this chapter, we considered a circuit which effects this transformation, where F is the quantum analgoue of our Boolean function f defining the set S.



4.6.1 Grover's Algorithm for any number of solutions

If we do not know k, we can easily rotate to a region of low probability. If we try to measure at every t, we would have to count up, since the state is destroyed each time you measure, and the number of steps is quadratic in t, which destroys the quantum advantage. Likewise, if we try to preserve the states by copying, we need a quadratic number of ancilliary qubits.

Suppose we randomly choose $\theta \in [0, 2\pi]$. There is a 50% chance that θ within 45 degrees of the y-axis. We can see this by cutting the circle into four equal parts using two diagonal lines, $y = x$ and $y = -x$. If θ is in the north or south cone, then $\sin^2(\theta) \geq \frac{1}{2}$. This gives an overall chance of success of 25%.

So could we just choose t uniformly from $1 \leq t \leq \sqrt{N}$? The problem is that if $k \geq N/2$, which means $\alpha \geq \frac{\pi}{4}$, then the number of iterations t we would choose is less than one. This means we are really not sampling θ uniformly. We fix this by guessing at the beginning. If $\alpha \geq \frac{\pi}{6}$, our guess succeeds with probability $\frac{1}{4}$, corresponding to $k \geq N/4$. Now assume that $\frac{1}{\sqrt{N}} \leq \alpha \leq \frac{\pi}{6}$, which corresponds to $1 \leq k \leq \frac{N}{4}$. We will choose t uniformly from 1 to $\sqrt{\frac{N}{4}}$. So what is the rotation $t \cdot 2\alpha$?

Suppose $2\alpha = \frac{2}{\sqrt{N}}$. Then θ ranges from $\frac{2}{\sqrt{N}}$ to 1. Whereas if $2\alpha = \frac{\pi}{3}$, then t ranges from $\frac{\pi}{3}$ to $\sqrt{N}\frac{\pi}{6}$.

4.7 Measurement

Any measurement of the properties of a quantum mechanical system must obey certain strange rules, determined from experiment, that are not needed when

measuring properties of classical systems. These restrictions are subsumed under the heading, the Born Rule, named after Max Born from the University of Göttingen (Born 1926). In quantum mechanics, *observables*, or properties we can measure, are represented by Hermitian operators. Why would this be? Thinking back to (Hardy 2001), we know that for an N-dimensional quantum space, it takes N^2 pieces of information to determine the outcome of a measurement, which is exactly the number of free parameters in a Hermitian operator on the space. We also know that only N outcomes can be distinguished by the measurement, and these are identified with the N orthogonal eigenvectors of the operator [1]. Precisely, the Born Rule states that

1. the result of the measurement will be one of the eigenvalues λ_i of the Hermitian operator A,

2. the probability of measuring a given eigenvalue λ_i will equal $\mathrm{Tr}(P_i\rho)$, where P_i is the orthogonal projection onto the eigenspace of A corresponding to λ_i and ρ is the density operator for the state, and finally that

3. the system after the measurement will lie in the range of P_i.

In the simple case that A has a non-degenerate, discrete spectrum, the eigenspaces are one-dimensional, say spanned by $\hat{\phi}_i$, so that

$$\mathrm{Tr}(P_i\rho) = \mathrm{Tr}\Big(\hat{\phi}_i\hat{\phi}_i^\dagger\rho\Big) = \hat{\phi}_i^\dagger\rho\hat{\phi}_i.$$

If we are in a pure state ψ, then the density operator has rank one, and this expression becomes

$$\hat{\phi}_i^\dagger\psi\psi^\dagger\hat{\phi}_i = \Big|\psi^\dagger\hat{\phi}_i\Big|^2.$$

The naturalness of the Born Rule is questionable. Would other schemes work just as well and be less artificial? Andrew Gleason settled this question when he proved Gleason's Theorem, which states that all assignments of probabilities to unit vectors (or, equivalently, to the operators that project onto them) that satisfy certain criteria take the form of applying the Born rule to some density operator. First, we require that probabilities are associated with each unit vector in the Hilbert space in such a way that they sum to one for any set of unit vectors comprising an orthonormal basis. This is just an uncontroversial normalization. Second, the probability associated with a unit vector is a function of the density operator and the unit vector, and not of any additional information such as the choice of basis for that vector or the order of measurements. This requirement is called *noncontextuality*, and it is at the heart of the argument over no-go theorems for hidden variables.

[1] If the operator has a continuous spectrum, we can still identify outcomes with a projection-valued measure (find discussion of this and POVMs)

By *hidden variable theory*, we mean a deterministic theory which reproduces the experimental results of quantum mechanics, but where the statistical nature arises from our ignorance of other variables influencing the outcomes, not from indeterminacy in the theory. Einstein and Bell both believed strongly that the world works in this way. A *no-hidden-variables theorem*, or no-go theorem, would like to show that these theories are impossible. However, we already know that the hidden-variables theory of David Bohm reproduces all the results of quantum mechanics. Thus the best we can hope for is to outlaw hidden-variable theories which have certain characteristics, such as being local or noncontextual.

Noncontextuality, the idea that making a measurement should be independent of the other measurements I make for a set of commuting operators, seems reasonable on its face. However, Bell points out that when measuring A in a set of commuting operators A, B, C, I would likely have a different experimental setup than measuring A in A, F, G. In the most famous and successful hidden variable theory, Bohmian mechanics, the theory is explicitly nonlocal which makes it noncontextual. In (Mermin 1993), we summarizes the theory beautifully

> The wave function guides the particles like this: each particle obeys a first order equation of motion specifying that its velocity is proportional to the gradient with respect to its position coordinates of the phase of the N-particle wave function, *evaluated at the instantaneous positions of all the other particles*. It is the italicized phrase which is responsible for the "hideous" non-locality whenever the wave function is correlated. If the wave function factors then the phase is a sum of phases associated with the individual particles and the non-locality goes away.

The *Bell-Kochen-Specker Theorem* is a simple demonstration that, for some collection of operators (or you can think of them as directions), there is no assignment of probabilities that will preserve the summability and noncontextuality conditions. Mermin has an elegant proof of this for a four dimensional system. He also points out that noncontextuality can be converted to locality by using a special input state. If the input is actually the tensor product state consisting of two particles, which can then be separated by a space-like interval, then the violation of noncontextuality is also a violation of locality, as he points out in the quote above.

As (Hardy 2001) shows, a crucial difference between the quantum and classical settings is that the dimension of the our state space N is less than the number of degrees of freedom K, in fact $K = N^2$. Hardy shows that this is a possible outcome of his axioms. The dimension N of the state space is the maximum number of states that can be reliably distinguished from one another in a single shot measurement. What this means is that we have a collection of states $\{|\psi_i\rangle\}$ and a set of commuting observables $\{O_i\}$ such that each observable will *always* pick out its corresponding state, so that we can do one measurement

and know for sure which state we had. We can state this formally as

$$\langle\psi_i|O_i|\psi_i\rangle = 1 \tag{4.90}$$

$$\sum_i O_i = 1 \tag{4.91}$$

since the observables exhaust the space. Now

$$\begin{aligned}
\langle\psi_1|\psi_1\rangle &= 1 \\
\langle\psi_1|\sum_i O_i|\psi_1\rangle &= 1 \\
\sum_i \langle\psi_1|O_i|\psi_1\rangle &= 1 \\
\langle\psi_1|O_1|\psi_1\rangle + \sum_{i\neq 1} \langle\psi_1|O_i|\psi_1\rangle &= 1 \\
1 + \sum_{i\neq 1} \langle\psi_1|O_i|\psi_1\rangle &= 1 \\
\langle\psi_1|O_i|\psi_1\rangle &= 0 \forall i \neq 1 \\
\sqrt{O_i}\psi_1 &= 0
\end{aligned}$$

so that each observable must annihilate all other states. Suppose we decompose $|\psi_2\rangle = \alpha|\psi_1\rangle + \beta|\phi\rangle$ where $|\phi\rangle$ is orthogonal to $|\psi_1\rangle$ and $|alpha|^2 + |\beta|^2 = 1$. Now if $|\psi_1\rangle$ is not orthogonal to $|\psi_2\rangle$ then $|beta| < 1$. However, this is a problem since

$$\langle\psi_2|O_2|\psi_2\rangle = |\beta|^2 \langle\phi|O_2|\phi\rangle \tag{4.92}$$

$$\leq |\beta|^2 \tag{4.93}$$

$$< 1 \tag{4.94}$$

which contradicts our assumption that this measurement would succeed with probability one. This means that only orthogonal states can be reliably distinguished this way, and is a consequence of the linearity of measurement.

4.8 Thoughts on Quantum Weirdness

It may not be the fact that a thing can be in a superposition of states that is weird. It sounds weird if your states are "alive" and "dead", but it sounds perfectly normal if you are in a superposition of red and blue (purple). Our common experience is filled with objects in a mixed state. The weird thing is that when I make a certain measurement of this thing, I can only get out one of the initial states, red or blue, but not purple. However I could make a different measurement and only get purple or cyan, and not red or blue. Thus it seems that the structure of the theory is not weird, it's the act of

measurement. In addition, Hardy argues (Hardy 2001) that this superposition is really a consequence of demanding continuity among the pure states.

Measurement is not really a single act, since quantum systems "measure" themselves all the time by interacting with other quantum systems. As long as we can fully describe the composite system with quantum mechanics, it's not a "measurement". Really, it seems that measurement is the act of bringing a quantum system into equilibrium with another very large quantum system, which destroys correlations it might have had with other systems. I think this is "decoherence". So the question is, how can this kind of equilibration eliminate all states but the eigenstates of the operator. Suppose the basis states are eigenfunctions of a quantum operator, and the decoherence process involves applying that operator over and over again until we asymptote to fixed point. This kind of power iteration would drive the system to the largest eigenvector represented in the initial state. The analogy is not perfect, since we would need to drive the system to some state depending on the amplitude of that basis vector, but this kind of process could produce the weird behavior of "collapsing" the linear combination to a single basis state.

4.9 Problems

Problem IV.1 QALA 7.1

Problem IV.2 QALA 7.4

Problem IV.3 QALA 7.5

Problem IV.4 QALA 7.6

Problem IV.5 QALA 8.1

Problem IV.6 QALA 8.2

Problem IV.7 QALA 8.4

Problem IV.8 QALA 9.1

Problem IV.9 QALA 9.2

Problem IV.10 QALA 13.1

Problem IV.11 QALA 13.2

Problem IV.12 QALA 13.3

Problem IV.13 QALA 13.5

References

Deutsch, David and Richard Jozsa (1992). "Rapid solution of problems by quantum computation". In: *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439.1907, pp. 553–558.

Born, Max (1926). "Zur Quantenmechanik der Stoßvorgänge". In: *Zeitschrift für Physik* 37.12, pp. 863–867. DOI: 10.1007/BF01397477.

Hardy, Lucien (2001). "Quantum theory from five reasonable axioms". In: eprint: quant-ph/0101012.

Mermin, N. David (1993). "Hidden variables and the two theorems of John Bell". In: *Reviews of Modern Physics* 65.3, p. 803.

Chapter 5

Problem Solutions

5.1 Introduction

Index