Some results in quantum learning theory

Faris Sbahi

Contents

Chapter 1

Introduction

In this document, we provide a presentation of the latest results in quantum learning theory alongside theoretical and empirical extensions.

The first part of our paper is a review: First, we present an overview of learning theory and the mathematical methods necessary for our analyses (whereas the Appendix provides the necessary tools of quantum information theory for the unfamiliar reader). Next, we present a review of the brief history of quantum machine learning. The subsequent part of our paper consists of analyses and extensions of recent results in quantum learning theory: (1) supervised learning using hybrid quantum-classical circuits, (2) Tang's [40] idea of least-square sampling providing parallel classical algorithms for quantum machine learning algorithms that solve singular value transformation problems, and (3) information theoretic bounds on quantum computational learning.

In the first part of our paper, we extend quantum feature maps which seek to solve the quantum encoding problem by encoding data inputs into a quantum state that implicitly performs the feature map given by a kernel function. Therefore, if the kernel is sufficiently difficult to evaluate classically, then there may exist a quantum advantage. Our contribution comes in two parts: (1) we offer a theoretical extension which identifies binary classification with state identification for $m = 2$ density matrices. Hence, we propose using the trace distance as a measure to tune hyper-parameters which determine the non-linearity of our feature map, and (2) we evaluate the performance of quantum feature maps using variational circuits on practical datasets, in addition to reproducing the results of [22] where they considered kernel-separable data.

In the next part, we apply the methods of Tang to an analysis of a polynomial singular value transformation. In other words, given a low-rank matrix $A \in \mathbb{R}^{m \times n}$ with SVD $\sum_i \sigma_i |u_i\rangle \langle v_i|$ and a vector $b \in \mathbb{R}^n$, assume that we have ℓ^2 sample and query access to b. Then, we can approximately simulate sample and query access to $\sum_i (\sigma_i)^m |u_i\rangle \langle v_i| b$ for any $m \in \mathbb{Z}$.

In the final part, we present the bounds due to [5, 6] by exposing a general method introduced in these works which utilizes principles of quantum information theory to derive sample complexity bounds on various concept classes, oracle access, and learning settings.

Chapter 2

Learning Theory and Mathematical Methods

Machine learning explores the study and construction of algorithms that are capable of finding patterns in data. Hence, this entails studying the ability of learning models to capture these patterns and generalize, as one would in statistical learning theory, in addition to studying model efficiency, enabled by computational complexity theory.

It is often natural to describe these patterns as unknown functions, as one does in computational learning theory. For example, if we are learning a classification function (say handwritten digit classification), then our function acts on some space of examples (say the handwriting in terms of pixels) and returns a label (the corresponding digit). Hence, the first two sections of this chapter are dedicated to tools and formalism for this view of machine learning.

On the other hand, many of the widely used supervised and unsupervised machine learning algorithms amount to linear algebraic data analysis techniques. For example, consider that ordinary least squares regression and ridge regression can be given in terms of a closed-form product of matrices and matrix inverses. Similarly, principal component analysis on a feature matrix A is essentially solving for the largest-eigenvalue eigenvectors of its singular matrix $A^\dagger A$. For some machine learning algorithms, such as SVM, the most computationally expensive steps involve solving linear and nonlinear convex programs. Nevertheless, we often rely on linear algebraic subroutines to solve these optimization problems.

We will show in Section 5 that it is reasonable to compare quantum algorithms with quantum state preparation to classical algorithms with sample and query access to input. Hence, we survey ideas from randomized linear algebra that rely on this sort of sample and query access. Then, in Section 5 we will use these concepts to show that many quantum machine-learning algorithms that solve linear algebra problems in low-dimensional spaces have a classical, randomized analog.

2.1 Boolean Fourier Analysis

A boolean function is a function

$$f : \{0,1\}^n \to \{0,1\}$$

or by relabeling

$$f : \{-1, +1\}^n \to \{-1, +1\}$$

The domain of a boolean function is the "Hamming cube", $\mathbb{B}^n$.

Definition 2.1.1. *Let $S \subseteq [n]$ with $n \in \mathbb{Z}$. Then,*

$$\chi_S(x) := \prod_{i \in S} x_i \qquad \text{(with } x^{\emptyset} = 1 \text{ by convention)}$$

Hence, it is clear that we can view $\chi_S(x) : \{-1, +1\}^n \to \{-1, +1\}$ as computing the parity of x over the set S.

Theorem 2.1.2. *Every function $f : \{-1, +1\} \to \mathbb{R}$ can be uniquely expressed as a multilinear polynomial,*

$$f(x) = \sum_{S \subseteq [n]} \hat{f}(S)\chi_S(x)$$

We denote this expansion as the Fourier expansion of f, and term the real coefficent $\hat{f}(S)$ the Fourier coefficient of f on S.

Definition 2.1.3. *Define an inner product $\langle \cdot, \cdot \rangle$ on $f, g : \{-1, +1\} \to \mathbb{R}$ by*

$$\begin{aligned} \langle f, g \rangle &= 2^{-n} \sum_{x \in \{-1,+1\}^n} f(x)g(x) \\ &= E_{x \sim \{-1,+1\}^n}[f(x)g(x)] \end{aligned}$$

where $x \sim \{-1, +1\}^n$ denotes that x is chosen uniformly at random from $\{-1, +1\}^n$.

Theorem 2.1.4. *The set of all functions $f : \{-1, +1\}^n \to \mathbb{R}$ forms a vector space V such that $\dim V = 2^n$. The $\{\chi_S\}, S \subseteq [n]$ form an orthonormal basis for V.*

Proof. Let f, g both map $\{-1, +1\}^n$ to $\mathbb{R}$. Clearly, by Theorem 2.1.2, $h = \alpha f + g$ for $\alpha \in \mathbb{R}$ is also a boolean function $h : \{-1, +1\}^n \to \mathbb{R}$. Hence, we have a vector space V of boolean functions and $\dim V = 2^n$ because we can describe a function by how it acts on each point of the Hamming cube. So, we can canonically identify $V \cong \mathbb{R}^{2^n}$. Because the number of parity functions is 2^n and they are independent, we conclude that the χ_S form a basis of V.

If this basis is indeed orthonormal, then the Definition above requires

$$\langle \chi_S, \chi_T \rangle = \begin{cases} 1, S = T \\ 0, S \neq T \end{cases}$$

So, observe that

$$\begin{aligned}\chi_S(x)\chi_T(x) &= \prod_{i\in S}\prod_{j\in T} x_i x_j \\ &= \prod_{i\in S\cap T} x_i^2 \prod_{j\in S\Delta T} x_i \\ &= \chi_{S\Delta T}(X)\end{aligned}$$

where $S\Delta T$ is the symmetric difference $(S\cup T)\setminus(S\cap T)$.

Now, we claim that $E[\chi_{S\Delta T}] = 1$ iff $S\Delta T = \emptyset$ and 0 otherwise. Indeed, we've defined that $\chi_\emptyset = 1$ so $E[\chi_\emptyset] = 1$. Otherwise,

$$E[\prod_{i\in S\Delta T} x_i] = \prod_{i\in S\Delta T} E[x_i]$$

by independence of the $\{x_i\}$. But each $E[x_i] = (1/2)(+1) + (1/2)(-1) = 0$. □

Proposition 2.1.5. *For $f : \{-1,+1\}^n \to \mathbb{R}$ and $S \subseteq [n]$, the Fourier coefficient of f on S is given by*

$$\hat{f}(S) = \langle f, \chi_S\rangle = E_{x\sim\{-1,+1\}^n}[f(x)\chi_S(x)]$$

Theorem 2.1.6. *For any $f, g : \{-1,+1\} \to \mathbb{R}$,*

$$\langle f, g\rangle = E_{x\sim\{-1,+1\}^n}[f(x)g(x)] = \sum_{S\in[n]} \hat{f}(S)\hat{g}(S) \qquad \text{(Parseval's Theorem)}$$

Hence,

$$\langle f, f\rangle = \sum_{S\in[n]} (\hat{f}(S))^2 = 1 \qquad \text{(Plancherel's Theorem)}$$

2.2 Computational Learning Theory

Computational learning theory addresses the following task: Given a source of "examples" $(x, f(x))$ from an unknown function f, compute a "hypothesis" function h that is good at predicting $f(y)$ on future inputs y.

Definition 2.2.1. *In the PAC ("probably approximately correct") model learning under the uniform distribution on $\{-1,+1\}^n$, a learning problem is identified with a concept class $\mathcal{C}$, which is just a collection of functions $f : \{-1,+1\}^n \to \{-1,+1\}$. A learning algorithm $\mathcal{A}$ for $\mathcal{C}$ is a randomized algorithm which has limited access to an unknown target function $f \in \mathcal{C}$. These two access models, in increasing order of strength are:*

- *"random examples", meaning $\mathcal{A}$ can draw pairs $(x, f(x))$ where $x \in \{-1,+1\}^n$ is uniformly random*

- *"queries", meaning $\mathcal{A}$ can request the value $f(x)$ for any $x \in \{-1,+1\}^n$ of its choice.*

In addition $\mathcal{A}$ is given an accuracy parameter $\epsilon \in [0, 1/2]$ and failure parameter δ. The output of $\mathcal{A}$ is required to be a hypothesis function $h : \{-1,+1\} \to \{-1,+1\}$. We say that A learns $\mathcal{C}$ with error ϵ if for any $f \in \mathcal{C}$, $\mathcal{A}$ outputs an h which is ϵ-close to f, i.e.

$$dist(f,h) \leq \epsilon$$

with probability $\geq 1 - \delta$.
Note that dist is the relative Hamming distance more generally defined as total variation distance (2.3.1).

Clearly, we can learn any such f in $O(2^n)$ time for $\epsilon = 0$ (there are only 2^n hypotheses). If $\mathcal{C}$ is complex, then exponential running time may be the best we can do. Nevertheless, if $\mathcal{C}$ contains relatively "simple" functions then improved bounds are possible. A common way of doing so is discovering "most of" a function's Fourier spectrum.

Definition 2.2.2. *Let $\mathfrak{F}$ be a collection of subsets $S \subseteq [n]$. We say that the Fourier spectrum of $f : \{-1,+1\}^n \to \mathbb{R}$ is ϵ-concentrated on $\mathfrak{F}$ if*

$$\sum_{S \subseteq [n], S \notin \mathfrak{F}} \hat{f}(S)^2 \leq \epsilon$$

Using the notation of the previous section, we can equivalently write

$$Pr_{S \sim S_f}[S \notin \mathfrak{F}] \leq \epsilon$$

where $S \sim S_f$ implies sampling S with probability $\hat{f}(S)^2$.

Proposition 2.2.3. *Given access to random examples from $f : \{-1,+1\}^n \to \{-1,+1\}$, there is a randomized algorithm which takes as input $S \subseteq [n], 0 < \delta, \epsilon \leq 1/2$, and outputs an estimate $\tilde{f}(S)$ for $\hat{f}(S)$ that satisfies*

$$|\tilde{f}(S) - \hat{f}(S)| \leq \epsilon$$

with probability $\geq 1 - \delta$ in running time $poly(n, 1/\epsilon) \cdot \log(1/\delta)$.

Proof. First, we know that $\hat{f}(S) = \langle f(x), \chi_S(x) \rangle$, recalling that this is an expectation across all $x \in \{-1,+1\}^n$ from the previous section. Hence, given an example $(x_1, f(x_1))$, we can compute $f(x_1)\chi_S(x_1) \in \{-1,+1\}$ to empirically estimate $\hat{f}(S)$. From here, we simply apply a Chernoff bound (Corollary 2.3.4.1) to determine the number of samples required to satisfy our claim.

Say we are given random samples $x_1, \cdots, x_s$. Then, write random variable $X_i = f(x_i)\chi_S(x_i)$ and $X := \sum_i X_i$. Then,

$$\Pr(X - E[X] \geq \epsilon) \leq 2e^{-\frac{s\epsilon^2}{2}}$$

Hence, in order to satisfy that this probability is $\leq \delta$, we require $s = O(\frac{1}{\epsilon^2} \log(1/\delta))$ □

Proposition 2.2.4. *Suppose that $f : \{-1,+1\}^n \to \{-1,+1\}$ and $g : \{-1,+1\}^n \to \mathbb{R}$ satisfy $\|f-g\|_2^2 \leq \epsilon$. Let $h : \{-1,+1\}^n \to \{-1,+1\}$ be defined by $h(x) = sgn(g(x))$. Then,*

$$dist(f,h) \leq \epsilon$$

Proof. Since $|f(x)-g(x)|^2 \geq 1$ whenever $f(x) \neq sgn(g(x))$, we conclude

$$\begin{aligned} dist(f,h) &= Pr_x[f(x) \neq h(x)] \\ &= E_x[1_{f(x)\neq sgn(g(x))}] \\ &\leq E_x[|f(x)-g(x)|^2] = \|f-g\|_2^2 \end{aligned}$$

□

Theorem 2.2.5. *Assume learning algorithm $\mathcal{A}$ has (at least) random sample access to target $f : \{-1,+1\}^n \to \{-1,+1\}$. Suppose that $\mathcal{A}$ can identify a collection $\mathfrak{F}$ of subsets on which f's Fourier spectrum is $\epsilon/2$-concentrated. Then using $poly(|\mathfrak{F}|, n, 1/\epsilon)$ additional time, $\mathcal{A}$ can with high probability output a hypothesis h that is ϵ-close to f*

Proof. For each $S \in \mathfrak{F}$ the algorithm uses Proposition 2.2.3 to produce an estimate $\tilde{f}(S)$ for $\hat{f}(S)$ which satisfies $|\hat{f}(S)-\tilde{f}(S)| \leq \frac{\sqrt{\epsilon}}{2\sqrt{|\mathfrak{F}|}}$ except with probability at most $1/(10|\mathfrak{F}|)$. Overall this requires $poly(|\mathfrak{F}|, n, 1/\epsilon)$ time and all $|\mathfrak{F}|$ estimates have the desired accuracy with probability $\geq 9/10$.

Now, $\mathcal{A}$ forms the real-valued function $g = \sum_{S\in\mathfrak{F}} \tilde{f}(S)\chi_S$ and outputs hypothesis $h = sgn(g)$. By Proposition 2.2.4, it suffices to show that $\|f-g\|_2^2 \leq \epsilon$. Indeed,

$$\begin{aligned} \|f-g\|_2^2 &= \sum_{S\subseteq[n]} \widehat{f-g}(S)^2 && \text{(Parseval's Theorem)} \\ &= \sum_{S\in\mathfrak{F}} (\hat{f}(S)-\tilde{f}(S))^2 + \sum_{S\notin\mathfrak{F}} \hat{f}(S)^2 \\ &\leq \sum_{S\in\mathfrak{F}} (\frac{\sqrt{\epsilon}}{2\sqrt{|\mathfrak{F}|}})^2 + \epsilon/2 && \text{(concentration assumption)} \\ &= \epsilon/4 + \epsilon/2 \leq \epsilon \end{aligned}$$

as desired. This theorem essentially reduces the algorithmic task of learning f to the algorithmic task of identifying a collection $\mathfrak{F}$ on which f's Fourier spectrum is concentrated. □

We've seen that the Fourier spectrum concentration provides a useful notion of the complexity of a *function*. In a more general sense, we can describe the complexity of a *concept class* in terms of a combinatorial parameter known as the VC dimension.

Definition 2.2.6. *(VC Dimension) Let $S_1, \cdots, S_{2^n} \subseteq \{0,1\}^n$ give all possible binary labelings of a dataset of size n. If for each S_i there is a $c_i \in \mathcal{C}$ which assigns the dataset this labelling, then S is said to be shattered by $\mathcal{C}$. The VC dimension of $\mathcal{C}$ is the size n of the largest dataset which is shattered by $\mathcal{C}$.*

Example 2.2.7. *The VC dimension of halfspaces in $\mathbb{R}^p$ is $p+1$. To see this, note that the definition of VC dimension asks us to find any dataset in our domain with size $p+1$. In which case, we can choose a dataset with points $\{e_i\}_{1\leq i\leq d}$ where e_i is the standard basis vector with a 1 in the ith position and 0 elsewhere and $e_0 := 0$.*

A halfspace is specified by a vector $w \in \mathbb{R}^p$ and decision rule

$$f(x) = sgn(w \cdot x + b)$$

Because f is a boolean function, we can decompose it in the Fourier basis and simply give positive Fourier coefficient to each e_i that S_i specifies has label $+1$ and negative coefficient otherwise. The bias b is determined by the label of $e_0 = 0$.

2.3 Randomized Linear Algebra

2.3.1 Probability Bounds

Definition 2.3.1. *Total Variation Distance.*

Let P and Q be distinct probability measures on a σ-algebra $\mathcal{F}$ of subsets of the sample space Ω. Then, the total variation distance is given by

$$\delta(P, Q) = \sup_{A\in\mathcal{F}} |P(A) - Q(A)|$$

Proposition 2.3.2. *(Markov's Inequality) Let X be a nonnegative random variable. Then,*

$$Pr(X \geq t) \leq \frac{1}{t}E[X]$$

for all $t \geq 0$.

Proposition 2.3.3. *(Chebyshev's Inequality) Let X be a random variable with finite variance $Var(X)$. Then,*

$$Pr(|X - E[X]| \geq t) \leq \frac{1}{t^2}Var[X]$$

for all $t \geq 0.x$

Proposition 2.3.4. *(Hoeffding–Chernoff Inequality)*

Let $X_1, X_2, \cdots, X_s$ be i.i.d real random variables. For any positive, real numbers a, t we have that, from Markov's inequality,

$$\begin{aligned}\Pr\left(\sum_{i=1}^{s} X_i \geq a\right) &\leq e^{-ta}E\left[\prod_{i=1}^{s} e^{tX_i}\right] \\ &= e^{-ta}\prod_{i=1}^{s} E\left[e^{tX_i}\right]\end{aligned}$$

by independence.

Corollary 2.3.4.1. *(Chernoff Bounds) Let $X_1, X_2, \cdots, X_s$ be i.i.d real random variables such that $X_i \in [a_i, b_i]$ with probability 1. Define $X := \sum_{i=1}^{s} X_i$. Then,*

$$\Pr(|X - E[X]| \geq t) \leq 2e^{-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}}$$

Example 2.3.5. *Define $H_x(p)$ to be an entropy function $p\log_x p + (1-p)\log_x(1-p)$. Similarly, define Then,*

$$\begin{aligned}\binom{n}{k} &\leq e^{nH_e(k/n)} \\ &= 2^{nH_2(e)(k/n)}\end{aligned}$$

for all $k \leq n$. Furthermore,

$$\begin{aligned}\sum_{i=1}^{m}\binom{n}{i} &\leq e^{nH_e(m/n)} \\ &= 2^{nH_2(e)(k/n)}\end{aligned}$$

for all $m \leq n/2$.

Theorem 2.3.6. *(Hoeffding–Chernoff Inequality for matrix-valued random variables) [23]*
Let X be a random variable taking values which are real symmetric $d \times d$ matrices. Suppose $X_1, X_2, \cdots, X_s$ are i.i.d. draws of X. For any positive real numbers a, t, we have

$$\Pr\left(\lambda_{\max}\Big(\sum_{i=1}^{s} X_i\Big) \geq a\right) \leq de^{-ta}\|E[e^{tX}]\|_2^s \tag{2.1}$$

$$\Pr\left(\Big\|\Big(\sum_{i=1}^{s} X_i\Big)\Big\|_2 \geq a\right) \leq de^{-ta}(\|E[e^{tX}]\|_2^s + \|E[e^{-tX}]\|_2^s) \tag{2.2}$$

where $\lambda_{\max}$ is the largest eigenvalue.

Proof. First, we can show that (2.1) $\Rightarrow$ (2.2). By definition of the 2-norm of a matrix,

$$\|\sum_i X_i\|_2 = \max\left(\lambda_{\max}\Big(\sum_i X_i\Big), \lambda_{\max}\Big(\sum_i (-X_i)\Big)\right)$$

since it is the square root of the maximum eigenvalue of $(\sum_i X_i^T)\sum_i X_i = (\sum_i X_i)\sum_i X_i$ and hence, equivalently, the maximum absolute value of an eigenvalue of X_i. Therefore, we can simply apply (2.1) to both X_i and $-X_i$ and we get (2.2).

So, we can focus our attention on (2.2). Let $S = \sum_i^s X_i$. Hence,

$$\lambda_{\max}(S) \geq a \Leftrightarrow \lambda_{\max}(tS) \geq ta$$

Furthermore, by considering the power series definition of the exponential,

$$\begin{aligned} &\Leftrightarrow \lambda_{\max}(e^{tS}) \geq e^{ta} \\ &\Rightarrow \operatorname{Tr}(e^{tS}) \geq e^{ta} \end{aligned}$$

since the trace is the sum of the matrix's eigenvalues. Since $\operatorname{Tr}(e^{tS}) \geq 0$, we can apply Markov's inequality

$$\Pr(\operatorname{Tr}(e^{tS}) \geq e^{ta}) \leq \frac{E[\operatorname{Tr}(e^{tS})]}{e^{ta}}$$

Now, we use the following lemma

Lemma 2.3.7. *Golden-Thompson Inequality*
If A and B are Hermitian matrices, then

$$\operatorname{Tr}(e^{A+B}) \leq \operatorname{Tr}(e^{A}e^{B})$$

□

Hence, we can let $A = t(\sum_i^{s-1} X_i)$ and $B = tX_s$. Then,

$$E_X\Big[\operatorname{Tr}(e^{tS})\Big] \leq E_X\Big[\operatorname{Tr}\Big(e^{t\left(\sum_i^{s-1} X_i\right)} e^{tX_s}\Big)\Big]$$

Since the expectation operator commutes with the summation of the trace by linearity of trace,

$$\begin{aligned} &= \operatorname{Tr}\left(E_X\left[e^{t\left(\sum_i^{s-1} X_i\right)} e^{tX_s}\right]\right) \\ &= \operatorname{Tr}\left(E_{X_1,X_2,\cdots,X_{s-1}}\left[e^{t\left(\sum_i^{s-1} X_i\right)}\right] E_{X_s}\left[e^{tX_s}\right]\right) \qquad \text{(by independence)} \end{aligned}$$

Now, we can apply Corollary (A.7.12.1), which gives

$$\begin{aligned} &\leq \operatorname{Tr}\left(E_{X_1,X_2,\cdots,X_{s-1}}\left[e^{t\left(\sum_i^{s-1} X_i\right)}\right]\right)\left\|E_{X_s}\left[e^{tX_s}\right]\right\|_2 \\ &= \operatorname{Tr}\left(E_X\left[e^{t\left(\sum_i^{s-1} X_i\right)}\right]\right)\left\|E_X\left[e^{tX}\right]\right\|_2 \\ &= E_X\left[\operatorname{Tr}\left(e^{t\left(\sum_i^{s-1} X_i\right)}\right)\right]\left\|E_X\left[e^{tX}\right]\right\|_2 \end{aligned}$$

So we can repeat this process iteratively, peeling an X_i each time from the left term. For clarity, the next step gives,

$$\begin{aligned} E_X\left[\operatorname{Tr}\left(e^{t\left(\sum_i^{s-1} X_i\right)}\right)\right] &\leq E_X\left[\operatorname{Tr}\left(e^{t\left(\sum_i^{s-2} X_i\right)} e^{tX_{s-1}}\right)\right] \\ &\leq E_X\left[\operatorname{Tr}\left(e^{t\left(\sum_i^{s-2} X_i\right)}\right)\right]\left\|E_X\left[e^{tX}\right]\right\|_2 \qquad \text{(applying (A.7.12.1) again)} \end{aligned}$$

Therefore, after peeling all terms but the last X_i, we have

$$E_X\Big[\operatorname{Tr}\big(e^{tS}\big)\Big] \leq E_X\Big[\operatorname{Tr}\Big(e^{tX}\Big)\Big]\Big\|E_X\Big[e^{tX}\Big]\Big\|_2^{s-1}$$

Hence, since the trace is the sum of eigenvalues, $\operatorname{Tr}\big(e^{tX}\big) \leq d\lambda_{\max}(e^{tX})$ i.e. the worst case of all d eigenvalues being the max

$$\leq d\Big\|E_X\Big[e^{tX}\Big]\Big\|_2^{s}$$

as desired. □

Lemma 2.3.8. *If $B \in \mathbb{C}^{d\times d}$ is a hermitian matrix for which $\|B\|_2 \leq 1$, then $e^B \leq I + B + B^2$*

Proof. We know that $e^{\lambda_i} \leq 1 + \lambda_i + \lambda_i^2$, $|\lambda_i|^2 \leq 1$. Hence,

$$e^{\lambda_i}\,|v_i\rangle\langle v_i| \leq (1 + \lambda_i + \lambda_i^2)\,|v_i\rangle\langle v_i|$$

where $|v_i\rangle$ is the corresponding eigenvector. This then implies

$$\begin{aligned} e^B &= \sum_{i=1}^{d} e^{\lambda_i}\,|v_i\rangle\langle v_i| \preceq \sum_{i=1}^{d}(1 + \lambda_i + \lambda_i^2)\,|v_i\rangle\langle v_i| \\ &= I + B + B^2 \end{aligned}$$

□

2.3.2 Computing Approximate Singular Vectors

Definition 2.3.9. *We have "query access" to $x \in \mathbb{C}^n$ if, given $i \in [n]$, we can efficiently compute x_i. We say that $x \in \mathcal{Q}$.*

Definition 2.3.10. *We have sample **and** query access to $x \in \mathbb{C}^n$ if*

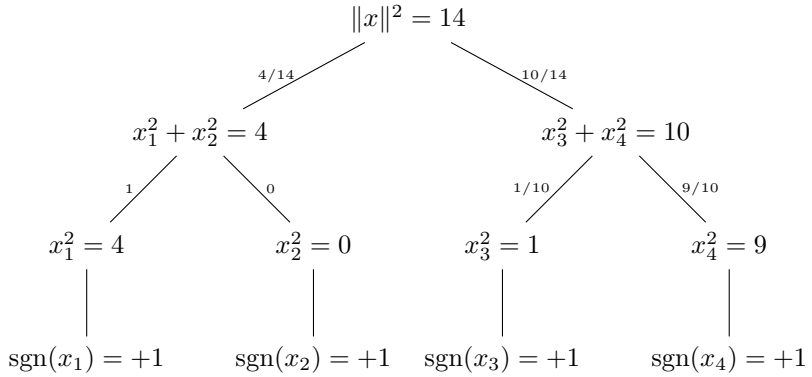
1. *We have query access to x i.e. $x \in \mathcal{Q}$ ($\Rightarrow \mathcal{SQ} \subset \mathcal{Q}$)*
2. *We can produce independent random samples $i \in [n]$ where we sample i with probability $|x_i|^2/\|x\|^2$ and can query for $\|x\|$.*

We say that $x \in \mathcal{SQ}$.

Definition 2.3.11. *For $A \in \mathbb{C}^{m\times n}$, $A \in \mathcal{SQ}$ (by abuse in notation) if*

1. *$A_i \in \mathcal{SQ}$ where A_i is the ith row of A*
2. *$\tilde{A} \in \mathcal{SQ}$ for $\tilde{A}$ the vector of row norms (so $\tilde{A}_i = \|A_i\|$).*

Example 2.3.12. *Say we have the vector $\vec{x} = (2, 0, 1, 3)$ and $\vec{x} \in \mathcal{SQ}$. Consider the following binary tree data structure.*



So, consider if we have length-square sampling access to the original matrix $A \in \mathbb{C}^{m \times n}$ i.e. $A \in \mathcal{SQ}$. Suppose we want to draw s rows in s i.i.d. trials. Then, pick row index i of A with probability

$$p_i = \frac{\|A_{(i,\cdot)}\|^2}{\|A\|_F^2} \tag{2.3}$$

and output random row

$$\begin{aligned} Y &= \frac{1}{\sqrt{sp_i}} \left\langle A_{(i,\cdot)} \right| \\ &= \frac{1}{\sqrt{s}} \frac{\|A\|_F}{\|A_{(i,\cdot)}\|} \left\langle A_{(i,\cdot)} \right| \end{aligned}$$

which is just a scaling of the ith row of A[1]. In other words,

$$\Pr\left(Y = \frac{1}{\sqrt{sp_i}} \left\langle A_{(i,\cdot)} \right| \right) = p_i$$

After sampling s rows, we implicitly define matrix R to be the concatenation of the outputted random rows. Therefore,

$$R = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_s \end{bmatrix} \in \mathbb{C}^{s \times n} \tag{2.4}$$

Note that $\langle Y_k|$ denotes the random row outputted by the procedure on the kth i.i.d. draw.

[1]The reason that we scale by s is so that the expectations of $A^\dagger A$ and $R^\dagger R$ coincide in the theorem that follows. The reason that we scale by p_i is so that the norms of all rows are equivalent—a fact which we'll utilize when we sample R column-wise.

Lemma 2.3.13. *Let $X = Y^\dagger Y - E[Y^\dagger Y]$ which evidently satisfies $E[X] = 0$. Then,*

$$E[X^2] \preceq E[(Y^\dagger Y)^2] \tag{2.5}$$

$$= A^\dagger A \|A\|_F^2 \frac{1}{s^2} \tag{2.6}$$

and so

$$\|E[X^2]\|_2 \leq \frac{1}{s^2}\|A\|_2^2\|A\|_F^2 \tag{2.7}$$

Furthermore,

$$\|X\|_2 = \frac{1}{s}\|A\|_F^2 \tag{2.8}$$

Proof. First, observe that $E[X^2]$ is the element-wise variance of $Y^\dagger Y$ and $E[(Y^\dagger Y)^2]$ is the corresponding second moment. Hence, the relation $E[X^2] \leq E[(Y^\dagger Y)^2]$ holds element-wise which implies that the matrix relation $E[X^2] \preceq E[(Y^\dagger Y)^2]$ holds as well.

Furthermore,

$$\begin{aligned}
E[(Y^\dagger Y)^2] &= \frac{1}{s^2}\sum_{i=1}^{m} \frac{p_i}{p_i^2} \left|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\middle|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\right| \\
&= \frac{1}{s^2}\sum_{i=1}^{m} \frac{\|A\|_F^2}{\left\langle A_{(i,\cdot)}\middle|A_{(i,\cdot)}\right\rangle} \left|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\middle|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\right| \qquad \text{(using (2.3))} \\
&= \frac{1}{s^2}\sum_{i=1}^{m} \|A\|_F^2 \left|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\right| = \frac{\|A\|_F^2}{s^2}\sum_{i=1}^{m} \left|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\right| \\
&= A^\dagger A \|A\|_F^2 \frac{1}{s^2}
\end{aligned}$$

Recall that $E[R^\dagger R] = \frac{1}{s}A^\dagger A$. So, observe that

$$\begin{aligned}
\|X\|_2 &= \|Y^\dagger Y - E[Y^\dagger Y]\|_2 \\
&= \frac{1}{s}\left\| \frac{1}{p_i} \left|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\right| - A^\dagger A \right\|_2 \\
&\leq \frac{1}{s}\max\left\{ \left\| \frac{1}{p_i} \left|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\right| \right\|_2, \left\|A^\dagger A\right\|_2 \right\} \\
&\leq \frac{1}{s}\max\left\{ \frac{\|A\|_F^2}{\| \left|A_{(i,\cdot)}\right\rangle \|_2^2} \left\| \left|A_{(i,\cdot)}\right\rangle \left\langle A_{(i,\cdot)}\right| \right\|_2, \|A^\dagger A\|_F^2 \right\}
\end{aligned}$$

using $\|A^\dagger A\|_2 = \|A\|_2^2 \leq \|A\|_F^2$ and plugging in (2.3).

$$= \frac{1}{s}\|A\|_F^2$$

□

Proposition 2.3.14. *If $t > 0, t \in \mathbb{C}$ satisfies $\|tX\|_2 \leq 1$ for all possible values of X, then*

$$\begin{aligned} \|E[e^{\pm tX}]\|_2 &\leq 1 + \frac{t^2}{s^2}\|A\|_2^2\|A\|_F^2 \\ &\leq e^{t^2\|A\|_2^2\|A\|_F^2/s^2} \end{aligned} \tag{2.9}$$

Proof. First, from (2.3.8) we know that $E[e^{tX}] \preceq E[I + X + X^2] = I + E[X^2]$ since $E[X] = 0$. Hence, we have the proposition by (2.7). □

Theorem 2.3.15. *Let $A \in \mathbb{C}^{m\times n}$ and $R \in \mathbb{C}^{r\times n}$ be constructed by the length-square sampling and scaling so that $E[R^\dagger R] = E[A^\dagger A]$ (requirements that are met by R defined in (2.4)). Then, for all $\epsilon \in [0, \|A\|/\|A\|_F]$[2], we have*

$$\Pr\left(\|R^\dagger R - A^\dagger A\| \geq \|A\|\|A\|_F\right) \leq 2ne^{\frac{-\epsilon^2 s}{4}}$$

Hence, for $s \geq (4\ln\frac{2n}{\eta})/\epsilon^2$, with probability at least $(1-\eta)$ we have

$$\|R^\dagger R - A^\dagger A\| \leq \epsilon\|A\|\|A\|_F$$

Proof. From our definition above, we have that

$$\begin{aligned} R^\dagger R &= \begin{bmatrix} Y_1^\dagger & Y_2^\dagger & \dots & Y_s^\dagger \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_s \end{bmatrix} \\ &= \sum_{k=1}^{s} |Y_k\rangle \langle Y_k| \end{aligned}$$

Let i_k give the index of the row sampled from A on the kth draw. Hence,

$$= \frac{1}{s}\sum_{k=1}^{s} \frac{1}{p_{i_k}} \left|A_{(i_k,\cdot)}\right\rangle \left\langle A_{(i_k,\cdot)}\right|$$

Furthermore,

$$\begin{aligned} E[R^\dagger R] &= \frac{1}{s}\sum_{k=1}^{s}\sum_{i_k=1}^{m} \frac{p_{i_k}}{p_{i_k}} \left|A_{(i_k,\cdot)}\right\rangle \left\langle A_{(i_k,\cdot)}\right| \\ &= \frac{1}{s}\sum_{k=1}^{s} A^\dagger A \\ &= A^\dagger A \end{aligned}$$

[2] If $\epsilon \geq \|A\|/\|A\|_F$, then we can simply use $\hat{0}$ to approximate $A^\dagger A$

Note that similarly $E[Y^\dagger Y] = \frac{1}{s}A^\dagger A$.

So, we can define $X_i = |Y_k\rangle\langle Y_k| - E\Big[\,|Y_k\rangle\langle Y_k|\,\Big]$ which is evidently an i.i.d. copy of X as we've defined previously. Hence,

$$\begin{aligned}\sum_{i=1}^{s} X_i &= R^\dagger R - E[R^\dagger R] \\ &= R^\dagger R - A^\dagger A\end{aligned}$$

Now, we can first apply Theorem 2.1 with $a = \epsilon\|A\|_2\|A\|_F$,

$$\begin{aligned}&\Pr\left(\left\|\Big(\sum_{i=1}^{s} X_i\Big)\right\|_2 \geq \epsilon\|A\|_2\|A\|_F\right) \\ &\qquad\leq ne^{-t\epsilon\|A\|_2\|A\|_F}(\|E[e^{tX}]\|_2^s + \|E[e^{-tX}]\|_2^s)\end{aligned}$$

for any $t > 0$. Hence, we can apply Proposition 2.3.14 which then gives us

$$\leq 2ne^{-t\epsilon\|A\|_2\|A\|_F}e^{t^2\|A\|_2^2\|A\|_F^2/s^2}$$

for $t \leq s/\|A\|_F^2$. Hence, we can set $t = \frac{\epsilon s}{2\|A\|_F\|A\|_2}$ (which is indeed less than $s/\|A\|_F^2$) and finally,

$$\leq 2ne^{-\epsilon^2 s/4}$$

Therefore, if we require that $s \geq (4\ln\frac{2n}{\eta})/\epsilon^2$ we then have

$$\begin{aligned}\Pr\left(\left\|\Big(\sum_{i=1}^{s} X_i\Big)\right\|_2 \geq \epsilon\|A\|_2\|A\|_F\right) &\leq 2ne^{-\frac{\epsilon^2}{4}\frac{4\ln\frac{2n}{\eta}}{\epsilon^2}} \\ &= 2ne^{-\ln\frac{2n}{\eta}} = \eta\end{aligned}$$

□

FKV Algorithm

Given $A \in \mathbb{C}^{m\times n}$ such that $A \in \mathcal{SQ}$ and some threshold k, we can output a description of a low-rank approximation of A with $\text{poly}(k)$ queries. Specifically, we output two matrices $S, \hat{U} \in \mathcal{SQ}$ where $S \in \mathbb{C}^{\ell\times n}$, $\hat{U} \in \mathbb{C}^{\ell\times k}$ ($\ell = \text{poly}(k, \frac{1}{\epsilon})$). This implicitly describes the low-rank approximation to A by the following theorem

Theorem 2.3.16. *Define*

$$\begin{aligned}D &:= A(S^\dagger\hat{U})(S^\dagger\hat{U})^\dagger && (\text{rank}(D) \leq k)\\ \sigma &:= \sqrt{2/k}\|A\|_F \\ A_\sigma &:= \sum_{\sigma_i\geq\sigma}\sigma_i|u_i\rangle\langle v_i| && (\text{using SVD})\end{aligned}$$

Then,

$$\|A - D\|_F^2 \leq \|A - A_\sigma\|_F^2 + \epsilon^2 \|A\|_F^2$$

with probability $\geq 1 - \delta$.

Proof. Sketch. Theorem 2.3.15 says that $\|S^\dagger S - A^\dagger A\| \leq \epsilon \|A\|_F^2$ with high probability ($\|A\|_2 \leq \|A\|_F$ in general). Similarly, by performing the sampling process on $S^\dagger$ to get $T \in \mathbb{C}^{\ell \times k}$, we have an analogous bound on $SS^\dagger - TT^\dagger$. Since T is a constant-sized matrix, we can compute $\hat{U}$, the large left singular vectors of T, which approximate the large left singular vectors of T. Then, $S^\dagger \hat{U}$ translates these large left singular vectors to their corresponding right singular vectors, which are the approximate singular vectors of A as desired. □

For more intuition, write SVDs

$$A = \sum_i |u_i\rangle \langle v_i|$$
$$S = \sum_i |w_i\rangle \langle \tilde{v}_i|$$
$$U = \sum_i |\tilde{w}_i\rangle \langle z_i|$$

where $\tilde{v}_i$ denotes that the right singular vectors of S well-approximate the right singular vectors of A and similarly for $\tilde{w}_i$. Hence,

$$UU^\dagger = \sum_i |\tilde{w}_i\rangle \langle \tilde{w}_i|$$
$$S^\dagger (UU^\dagger) S \approx \sum_i |\tilde{v}_i\rangle \langle \tilde{v}_i|$$
$$A(S^\dagger UU^\dagger S) \approx \sum_i |u_i\rangle \langle \tilde{v}_i|$$

To visualize the projection,

$$\begin{bmatrix} \cdots A \cdots \end{bmatrix} \begin{bmatrix} S^\dagger \\ \\ \end{bmatrix} \begin{bmatrix} \hat{U} \end{bmatrix} \begin{bmatrix} \hat{U}^\dagger \end{bmatrix} \begin{bmatrix} \cdots S \cdots \end{bmatrix}$$

Chapter 3

A Review of Quantum Machine Learning

3.1 Introduction

Since its conception, quantum computation and quantum information has taught us to "think physically about computation" [32]. Well, if quantum mechanics tells us that physical states are mathematically linear algebraic objects (vectors in a Hilbert space), then perhaps this lesson says that the type of computation best suited for quantum physical reality are linear algebraic problems, such as the machine learning ones noted above. This somewhat naive intuition turns out to have value, as we will show in this review.

Hence, this leads us to the idea of quantum machine learning which uses quantum algorithms as part of a larger implementation to outperform the corresponding classical learning algorithms.

Nevertheless, past linear algebraic analysis techniques, there exist other classes of machine learning algorithms such as deep learning built on artificial neural networks and reinforcement learning which models an environment as a Markov decision process [39]. Successes have been achieved in terms of (potential) quantum speedups in these arenas [14], but we won't explore these alternative machine learning routes further in our review.

In this chapter, we will cover the main theoretical results in terms of quantum algorithms relevant to quantum machine learning. Then, we'll describe specific learning problems that have achieved quantum speedups using these results. Finally, we'll discuss important limitations to these results which may pose substantive challenges to the future of quantum learning theory.

Our goal is to provide a birds-eye view of these concepts and refer the reader to appropriate references where they desire greater detail.

3.2 Comparing Machine Learning Performance

If we are to claim that some machine learning algorithms perform better on a quantum computer, we first must decide on a notion of "outperforming".

This is currently characterized by the advantage in runtime obtained by a quantum algorithm over the classical methods for the same task. We quantify the runtime with the asymptotic scaling

of the number of elementary operations used by the algorithm with respect to the size of the input, as one does in complexity theory.

The definition of an elementary operation is dependent on the choice of measure of complexity. Query complexity measures the number of queries to the information source for the classical or quantum algorithm. Hence, a quantum speedup results if the number of queries needed to solve a problem is lower for the quantum than for the classical algorithm [7].

3.3 Speedup Techniques

3.3.1 Solving Systems of Linear Equations

Solving linear systems of equations is a ubiquitous problem in machine learning. As we will discuss, many learning problems, such as least-squares regression and least-squares SVMs, require the inversion of a matrix. Hence, we will describe two common quantum algorithms which lead up to the recent HHL algorithm, named after the algorithm's authors, Harrow, Hassidim, and LLoyd.

HHL Algorithm

One such application of Phase Estimation (Section A.5.2) is with respect to solving linear systems of equations. This is the so-called HHL algorithm [26]. Here, we will cover the essential details of the algorithm.

The general problem statement of a linear system is if we are given matrix A and unit vector $\vec{b}$, then find $\vec{x}$ satisfying,

$$A\vec{x} = \vec{b}$$

However, assume that instead of solving for x itself, we instead solve for an expectation value $x^T Mx$ for some linear operator M. The original description of the algorithm provides runtime bound of $\tilde{O}(\log(N)\kappa^2 s^2/\epsilon)$, where s is sparsity measured by the maximum number of non-zero elements in a row and column of A, κ is the condition number, and ϵ is the approximation precision. Hence, we can only achieve speedup if the linear system is sparse and has a low condition number κ. Ambainis [3] and Childs [11] improved this depency on κ and s to linear and ϵ to poly-logarithmic.

This compares well considering that the best classical algorithm has a runtime of $O(N^{2.373})$ [13]. However, due to the large amount of pre-processing required, the algorithm is not used in practice. Standard methods, for example, based on QR-factorization take $O(N^3)$ steps [20].

So, assume that A in our linear system is an $N \times N$ Hermitian matrix. Notice that this is an "unrestrictive" constraint on A because we can always take non-Hermitian matrix A' and linear system $A'\vec{x} = \vec{b}$ and instead solve $\begin{bmatrix} 0 & A' \\ A'^\dagger & 0 \end{bmatrix} \begin{bmatrix} 0 \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$. Hence, we we will assume that A is Hermitian from here on.

Recall that because A is hermitian, then we can perform quantum phase estimation using e^{-iAt} as the unitary transformation. This can be done efficiently if A is sparse.

So, we first prepare $|b\rangle = \sum_i b_i |i\rangle$ (the representation of $\vec{b}$). We assume that this can be done efficiently or that $|b\rangle$ is supplied as an input.

Denote by $|\psi_j\rangle$ the eigenvectors of A with associated eigenvalues λ_j. Hence, we can express $|b\rangle$ as $|b\rangle = \sum_j \beta_j |\psi_j\rangle$. So, we initialize a first register to state $\sum_j \beta_j |\psi_j\rangle$ and second register to state $|0\rangle$. After applying phase estimation, we then have the joint state $\sum_j \beta_j |\psi_j\rangle \left|\widetilde{\lambda}_j\right\rangle$, where $\widetilde{\lambda}_j$ is an

approximation of λ_j. We'll assume that this approximation is perfect from here on, for the sake of demonstration.

Next we add an ancilla qubit and perform a rotation conditional on the first register which now holds $|\lambda_j\rangle$. The rotation transforms the system to

$$\sum_j \beta_j |\psi_j\rangle |\lambda_j\rangle \left(\sqrt{1 - \frac{C^2}{\lambda_j^2}} |0\rangle + \frac{C}{\lambda_j} |1\rangle\right)$$

for some small constant $C \in \mathbb{R}$ that is $O(1/\kappa)$.

Hence, we can undo phase estimation to restore the second register to $|0\rangle$.

Now, if we measure the ancillary qubit in the computational basis, we'll evidently collapse the state to $|1\rangle$ with some probability. We'd then have

$$\sum_j \frac{C}{\lambda_j} \beta_j |\psi_j\rangle |\lambda_j\rangle |1\rangle = C(A^{-1} |b\rangle)$$

In particular, the probability of getting this result is

$$\begin{aligned}
p(-1) &= \left(\sum_j \beta_j \langle\psi_j| \langle\lambda_j| \left(\sqrt{1 - \frac{C^2}{\lambda_j^2}} \langle 0| + \frac{C}{\lambda_j} \langle 1|\right)\right) |1\rangle \langle 1| \left(\sum_j \beta_j |\psi_j\rangle |\lambda_j\rangle \left(\sqrt{1 - \frac{C^2}{\lambda_j^2}} |0\rangle + \frac{C}{\lambda_j} |1\rangle\right)\right) \\
&= \sum_j \beta_j \langle\psi_j| \langle\lambda_j| \left(\sqrt{1 - \frac{C^2}{\lambda_j^2}} \langle 0| + \frac{C}{\lambda_j} \langle 1|\right) |1\rangle \langle 1| \beta_j |\psi_j\rangle |\lambda_j\rangle \left(\sqrt{1 - \frac{C^2}{\lambda_j^2}} |0\rangle + \frac{C}{\lambda_j} |1\rangle\right) \\
&= \sum_j \beta_j \langle\psi_j| \langle\lambda_j| \frac{C}{\lambda_j} \langle 1|1\rangle \langle 1| \beta_j |\psi_j\rangle |\lambda_j\rangle \frac{C}{\lambda_j} |1\rangle \\
&= \sum_j \beta_j^2 \frac{C^2}{\lambda_j^2} \\
&= \|A^{-1} |b\rangle\|^2 C^2 = \Omega(1/\kappa^2)
\end{aligned}$$

However, using amplitude amplification (Section A.6) this can be upper-bounded to $O(1/\kappa)$.

Finally, we can make a measurement M whose expectation value $\langle x| M |x\rangle$ corresponds to the feature of x we wish to evaluate.

The concessions we noted along the way clearly limit the algorithm's applicability to practical problems. We have three essential caveats to achieving exponential speedup: (1) A must be sparse and have a condition number that scales at most sublinearly with N, (2) $|b\rangle$ must be loaded in quantum superposition in $\log(N)$ time, and (3) $|x\rangle$ isn't actually be read out, but instead an expectation is computed.

Limitation (1) has been partially resolved by the work of [44] to achieve a quadratic speedup for dense matrices.

Limitation (2) may be solved by quantum RAM, which then has its own limitations discussed in the next section.

Limitation (3) is a general issue with regards to reading out classical information from a quantum state at the conclusion of a quantum algorithm because we would need at least N measurements to retrieve this classical data. Hence, this would eliminate the potential for exponential speed-up.

Hence, after observing these limitations we may wonder if there can exist a classical algorithm with the same caveats that can achieve the same runtime. Importantly, in the original paper the authors showed that HHL is universal for quantum computation. Hence, we can encode any quantum algorithm e.g. Shor's algorithm into a system of roughly 2^n linear equations in 2^n unknowns, and then use HHL to solve the system (i.e., simulate the algorithm) in polynomial time. Thus, provided we believe any quantum algorithm achieves an exponential speedup over the best possible classical algorithm, HHL can theoretically achieve such a speedup as well [1].

To conclude, HHL is a logarithmic time quantum algorithm for matrix inversion, a task arising in many learning problems. However, a number of caveats that include the requirement of a logarithmic access time to the memory and the impossibility of retrieving the solution vector with one measurement lead to the question of whether classical or parallel algorithms that make use of the same assumptions obtain similar, or better, runtimes in practice. Hence, we will need empirical data to further address this question.

3.3.2 Quantum Random Access Memory

The essence of machine learning is analyzing a vast amount of data. Hence, we must address the question of how classical data is encoded in quantum superposition, a concern brought up in our previous discussion of the matrix inversion algorithm.

So, assume that we have an classical vectors $\{v_1, \cdots, v_n : v_1 \in \mathbb{R}^m\}$ that need to be encoded for use as part of a quantum algorithm. Quantum random access memory (qRAM) can encode these classical vectors in superposition into $\log(nm)$ qubits in $\log(nm)$ time using its "bucket-brigade architecture"[19]. The basic idea is to use a three-level tree structure where the nm qubit "leaves" contain the entries of the n vectors in $\mathbb{R}^m$.

One of the central limitations of qRAM is that the number of resources scales as $O(nm)$ i.e. exponentially in the binary representation of n, m. There have been open questions of the viability of this model of memory, in practice. In particular, if each of the qubits must be error-corrected, then it seems entirely impractical for general use. Some of this concern has been answered by proponents who have showed that, given a certain error model, algorithms that require to query the memory a polynomial number of times (e.g. the HLL algorithm above) might not require fault-tolerant components. Still, the amplitude amplification algorithm above does require this error correction.

Furthermore, it may be only fair to compare qRAM to a parallel classical architecture, given that we are allowing resources to scale exponentially.

Considerable, too, is the fact that data must be distribute fairly uniformly over the quantum register or else qRAM is no longer efficient [1].

In conclusion, qRAM comes with a significant number of considerations in itself and hence should be subjected to empirical investigation to determine its genuine quantum speed-up in practice.

3.4 Applications

3.4.1 Principal Component Analysis

First, we consider the ubiquitous principal component analysis (PCA) algorithm. PCA reduces the dimensionality of data by transforming the features to uncorrelated weightings of the original features ("principal components"). This requires simply finding the eigenvalues and eigenvectors of the data covariance matrix.

Hence, let $v_j \in V$ where V is a d-dimensional vector space such that $d = 2^n = N$ and assume that $|v_j| = 1$ for simplicity. Hence, the covariance matrix of the data is given by $C = \sum_j v_j v_j^T$ whose eigenvalues and eigenvectors we seek to discover. So, suppose we can prepare quantum state $v_j \to |v_j\rangle$, choosing classical data vector v_j uniformly at random. For example, we can use qRAM, discussed above. Then, because of our random selection, the resulting quantum state has density matrix $\rho = \frac{1}{N} \sum_j |v_j\rangle \langle v_j|$ which we observe to be the covariance matrix, up to an overall factor.

Then, as Lloyd, Mohseni and Rebentrost describe [29], one way to perform quantum PCA use that given n copies of ρ we have the ability to apply the unitary operator $e^{-i\rho t}$ to any state σ with accuracy $O(t^2/n)$. This can be done by using repeated infinitesmal applications of the SWAP operator on $\rho \otimes \sigma$.

Hence, using $e^{-i\rho t}$, we can perform phase estimation on rho. So, let $|\psi_i\rangle$ be the eigenvectors of ρ with associated eigenvalues λ_i and so $\rho = \sum_j \lambda_j |\psi_j\rangle \langle \psi_j|$. Therefore, phase estimation gives us

$$\sum_j \lambda_j |\psi_j\rangle \langle \psi_j| \otimes \left|\tilde{\lambda_j}\right\rangle \left\langle\tilde{\lambda_j}\right|$$

where $\tilde{\lambda_j}$ is an approximation of λ_j. Hence, if we measure the second register and repeat this procedure several times, the most frequent outcome is the first principal component $|\psi_{max}\rangle$, and so on.

The algorithm works best when a small number of principal components have significantly greater eigenvalues than the rest. Hence, ρ is well represented by the subspace spanned by the principal components. In particular, let $m << d$ be the dimension of the subspace of principal components which represent ρ within some bound. Then, the algorithm has runtime $O(m \log n)$.

Hence, this gives a limitation: if the eigenvalues of the covariance are roughly equal and of size $O(1/d)$, then the algorithm reduces to scaling in time $O(d)$ (as does the classical algorithm). Furthermore, we evidently inherit the limitations of qRAM, if we use it for the state preparation above.

3.4.2 Support Vector Machines

Support vector machines seek to find an optimal (maximum margin) separating hyperplane between two classes of data in a dataset. Hence, if the training data is linearly separable, each class can be found on only one side of the hyperplane. In particular, let $\{x_1, \cdots, x_n\}$ be a set of observations where x_i is a d-dimensional vector with associated labels $y_i \in \{0, 1\}$. Hence, we seek a linear function $f(x) = \vec{w} \cdot \vec{x} - \vec{b}$ where $\vec{w}$ is a weight vector and $\vec{b}$ is a bias vector s.t. $y_i f(x_i) > 0, \forall i$. Furthermore, we seek to maximize the minimum $y_i f(x_i) > 0$, across all i.

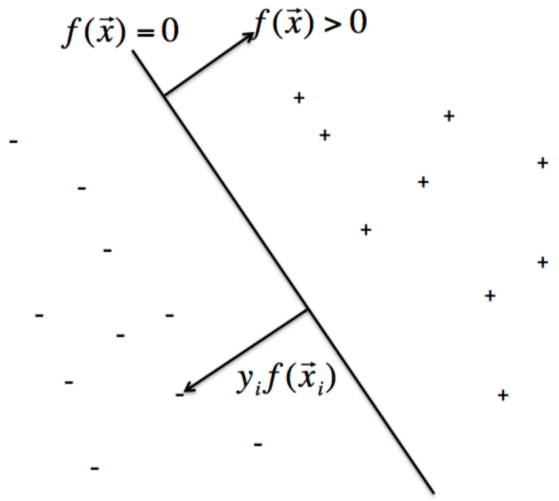


Figure 3.1: Visualization of a separating hyperplane and its associated margin taken from [36].

Much of the appeal of SVM is in its generalization to nonlinear hypersurfaces by replacing the constituent inner products with a kernel that implicitly maps the observations to another high dimensional space [36].

Solving SVM generally is performed by optimization of the dual problem which is a quadratic optimization problem. Hence, in [34], the authors show that quantum evaluation of inner products leads to exponential speed-ups in terms of d.

Full exponential improvements, however, can be achieved (with respect to d and n) in the case of least-squares SVMs [34]. In this case, finding a solution requires optimizing a least squares minimization. Hence, this type of minimization reduces to solving a set of linear equations. So, after data has been inputted efficiently (perhaps using qRAM), a modified version of the matrix inversion algorithm above can be used, incorporating methods from the PCA algorithm above to prepare the linear system.

The key point is, all the operations required to construct the maximum-margin separating hyperplane and to test whether a vector's classification can be performed in time that is polynomial in $\log n$.

3.4.3 Gaussian Processes

In [47], the authors demonstrate an application of HHL algorithm with respect to Gaussian process regression (GPR), another supervised learning method. In effect, GPR is a stochastic generalization of ordinary regression. In particular, let $\{x_1, \cdots, x_n\}$ be a set of observations where x_i is a d-dimensional vector with associated scalar targets y_i. In GPR, we consider the distribution over latent functions $f(x)$ which can return the correct labelling, y, given an inputted data-point. However, we further assume that this labelling is subject to Gaussian noise i.e. $f(x) = y + \epsilon$, where ϵ is the Gaussian noise. Hence, GPR uses Bayesian inference to return this distribution of latent functions such that they are consistent with the observed data.

Now, given this distribution and some test input x', we can predict its output by computing (1) a linear predictor known as the predictive mean and (2) the variance of the predictor with respect to x'. These values then give the distribution the y' that is consistent with training data. Hence, the quantum speedup comes from the fact that both of these values can be computed using a modification of the HHL algorithm, as described in [47]. Furthermore, the size of the output is independent from the size of the dataset which the authors use to show that, combined with the efficiency of HHL, can given exponential speedups in terms of d. Again, we require that the data can be loaded in efficiently initially e.g. using qRAM.

3.4.4 Optimization

Unsurprisingly, optimization methods are fundamental to many machine learning algorithms. Again, we can apply our quantum set of tools to several important optimization problems, including semi-definite programs (with potential super-polynomial speedups) and constraint satisfiability programs. Even easier to see, we can directly use our demonstrated results to solve quadratic programming problems which reduce to a linear system. If $N \times N$ matrices that define the linear system are sparse and low rank, we can use HHL and yield a solution in $\log N$ time–an exponential speedup.

Furthermore, iterative optimization, such as by means of gradient descent can be implemented by modifying PCA. In this case, several copies of a quantum state representing the solution are used to iteratively improve the solution. We may expect improvements in this type of optimization to lead to improvement in training neural networks on a quantum computer.

Interestingly, the quantum optimization algorithm finds approximate solutions for combinatorial optimization by "alternating qubit rotations with the application of the problem's penalty function" [7].

3.4.5 Topological Data Analysis

By now, we've seen that the presented algorithms suffer from a common limitation: the classical data must be loaded into a quantum superposition efficiently given that the speedups come from performing the computational steps after data is loaded in. However, this issue can be avoided in cases where the data points can be computed efficiently individually, as opposed to loading all of a large dataset in. In terms of machine learning, this can happen when the computation component of the algorithm requires exploring a comparatively small subset of the original dataset. In other words, if we can explore a subset of the input data to determine the distribution and other descriptive features of the overall data, we can then have the quantum algorithm generate the combinatorially larger space in quantum parallel, thereby computing the quantum dataset efficiently. This idea was used in the context of topological data analysis in [27].

Topological features, in particular, seem promising for this goal given that they are independent of the metric of choice and hence capture essential features of the data. In a discrete set, topological features are given by features that persist across spatial resolutions. These persistent features, formalized through persistent homology, are less likely to be artifacts of noise or parameterization. For example, the holes, voids, or connected components are examples of such features. The number of these types of features are defined for simplical complex (roughly a closed set of simplexes) as "Betti numbers".

In the paper, the authors show how to generate quantum states to encode the simplexes using logarithmically fewer qubits. Furthermore, they show that the Betti numbers can be efficiently represented using this representation. Important assumptions were made, in particular such that

the quantum state encoding the simplexes can be generated efficiently. One satisfying condition is if the number of simplexes in the complex are exponentially large. Hence, in at least some cases, we may extract exponential speed-ups in this type of topological analysis.

3.5 Challenges

The first obvious challenge to quantum machine learning is that the execution of quantum algorithms requires quantum hardware that does not exist at present.

Aside from this, as we've seen by now, many of the quantum algorithms which are used to potentially give quantum speedups come with several caveats and limitations to their use. As noted in [7], implicitly in [1], and repeatedly elsewhere, we can condense the general caveats into a few fundamental problems:

(1) The Input State Preparation Problem. The cost of reading in the input can in some cases dominate the cost of quantum algorithms. In many cases, we hoped qRAM would solve this problem. Understanding this factor is an important ongoing challenge.

(2) The Readout problem. Retrieving the solution in terms of classical information after performing quantum algorithms requires learning an exponential number of bits through repeated trials. We've shown that learning an expectation value of a linear operator can be a sound alternative (e.g. in the case of matrix inversion).

(3) The true cost problem. We require empirical study to show the number of gates or elementary operations to carry out a quantum algorithm, in practice. For example, we encountered this issue in the fundamental phase estimation algorithm (i.e. is it fair to assume that a U^{2^j} gate is an elementary operation?). Nevertheless, our query complexity bounds seem to indicate great advantages in specific cases, as we've noted.

Throughout this review, we've considered learning classical data which introduces problems (1) and (2), given that we have to interface between classical and quantum states. Hence, we may consider the case of applying quantum algorithms to quantum data. This is covered in detail in [2].

Chapter 4

Quantum Kernel Maps and Quantum Circuit Learning

We've discussed the classical machine learning algorithm known as Support Vector Machines (SVM) in Section 3.4.2. To reiterate, the goal is to construct a separating hyperplane such that the distance to the nearest training observation (minimum margin) is maximized. Much of the popularity of SVMs can be attributed to its association with the "kernel trick" which maps the data to a higher dimensional space so that it is separable or approximately separable.

More explicitly, let $\{x_i\}_{1 \leq i \leq n}$ be a set of observations where x_j is a d-dimensional vector in some inner product space Ω. These x_j have associated labels $y_i \in \{0, 1\}$. We seek a linear function $f(x) = \vec{w} \cdot \vec{x}$ where $\vec{w}$ is a weight vector s.t. $y_i f(x_i) > 0$ for all (or many) i. Furthermore, we seek to maximize the minimum $y_i f(x_i)$, across all i.

It turns out that this maximization can be restated as choosing w according to a convex program consisting of a linear combination of the all-pair Euclidean inner product $x_i \cdot x_j$. Hence, we may construct a feature map $\varphi : \Omega \to \mathcal{H}$ which maps the data to a higher dimensional Hilbert space $\mathcal{H}$. Then, instead, we consider the inner product $\langle x_i, x_j \rangle$ of $\mathcal{H}$ as part of the optimization procedure. This then results in a linear boundary in $\mathcal{H}$ which maps back to a nonlinear boundary in Ω. Therefore, the boundary is essentially tied to the inner product $\langle \varphi(x_i), \varphi(x_j) \rangle_{\mathcal{H}}$ which we term the "kernel".

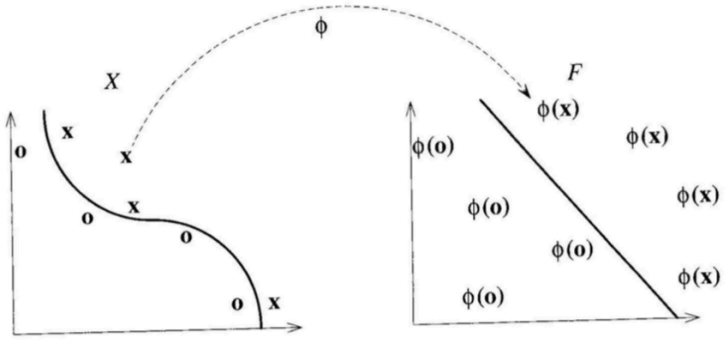


Figure 4.1: Nonlinear boundary created by kernel map. Data is mapped to a higher dimensional space where a linear boundary is created, and then this boundary is mapped back to the original space (creating the nonlinearity).

In our review of quantum machine learning, we discussed that input state preparation is a major challenge for practically achieving a quantum advantage with existing quantum machine learning algorithms. Well, one way to work around this problem is to tie input state preparation to a feature map that is difficult to evaluate classically. In other words, if we are interested in SVM (or any other machine learning algorithm) which leverages the "kernel" trick, then input data $\{x_i\}$ is only considered in terms of its feature map $\{\varphi(x_i)\}$ as part of the optimization process. If we can directly evaluate the feature map on classical inputs $\{x_i\}$ and if this feature map reaps a quantum advantage, then we've worked around the input state preparation problem.

4.1 Quantum Feature Maps

4.1.1 The Shifted Bent Map

Consider the feature map defined in [22].

First define,

$$U_{\Phi(x)} = \exp\Big(i \sum_{S\subseteq[n]} \varphi_S(x) \prod_{i\in S} Z_i\Big) \tag{4.1}$$

with nonlinear coefficients $\varphi_S(x) \in \mathbb{R}$. Then, the feature map is given by

$$\mathcal{U}_{\Phi(x)} = U_{\Phi(x)} H^{\otimes n} U_{\Phi(x)} H^{\otimes n} \tag{4.2}$$

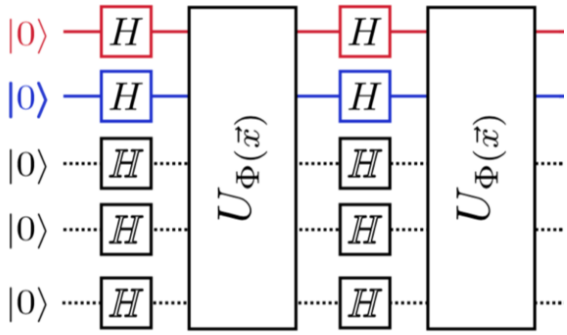


Figure 4.2: Figure from [22] of the feature map encoding circuit

Hardness Claim

The authors conjecture that it is hard to estimate this kernel up to an additive polynomially small error by classical means. The intuition for the conjecture is based in athe connection of this kernel map to a circuit family which solves the hidden shift problem for Maiorana-McFarland bent functions:

Definition 4.1.1. *(Bent functions) Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. We say that f is bent if the Fourier coefficients $\hat{f}(w) = \frac{1}{2^n}\sum_{x\in\{0,1\}}(-1)^{wx^t+f(x)}$ satisfy $|\hat{f}(w)| = 2^{-n/2}$, i. e., if the spectrum of f is flat.*

Recall the notion of ϵ-concentration of Fourier spectra discussed in Section 2.1. We described that this notion is useful in classical learning theory for describing a boolean function's complexity. Nevertheless, we'll find that bent functions which are in this sense the "most" complex, often are associated with problems with efficient quantum solutions.

Definition 4.1.2. *(Hidden shift problem). Let $n \geq 1$ and let O_f be an oracle which gives access to two Boolean functions $f, g : \{0,1\}^n \to \{0,1\}$ such that the following conditions hold: (i) f, and g are bent functions, and (ii) there exist $s \in \{0,1\}^n$ such that $g(x) = f(x+s)$ for all $x \in \{0,1\}^n$. We then say that O_f hides an instance of a shifted bent function problem for the bent function f and the hidden shift $s \in \{0,1\}^n$. If in addition to f and g the oracle also provides access to the dual bent function $\tilde{f}$, then we use the notation $O_{f,\tilde{f}}$ to indicate this potentially more powerful oracle.*

Theorem 4.1.3. *Let $O_{f,\tilde{f}}$ be an oracle that hides an instance of a shifted bent function problem for a function f and hidden shift s and provides access to the dual bent function $\tilde{f}$. Then there exists a polynomial time quantum algorithm $\mathcal{A}_1$ that computes s with zero error and makes two quantum queries to $O_{f,\tilde{f}}$*

Proof. Let $f : \{0,1\}^n \to \{0,1\}$ be the bent function. We have oracle access to the shifted function $g(x) = f(x+s)$ via the oracle, i. e., we can apply the map $|x\rangle|0\rangle \mapsto |0\rangle|f(x+s)\rangle$ where $s \in \{0,1\}^n$ is the unknown string.

Recall that whenever we have a function implemented as $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$, we can also compute f into the phase as $U_f : |x\rangle \mapsto (-1)^{f(x)}|x\rangle$ by applying f to a qubit initialized in $|-\rangle$. The hidden shift problem is solved by the following algorithm $\mathcal{A}_1$:

1. Prepare the initial state $|0\rangle$
2. Apply Hadamard $H^{\otimes n}$ to prepare an equal superposition of all inputs
3. Compute the shifted function g into the phase to get

$$\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} (-1)^{f(x+s)} |x\rangle$$

4. Apply $H^{\otimes n}$ to get

$$\sum_w (-1)^{sw^t} \hat{f}(w) |w\rangle = \frac{1}{2^{n/2}} \sum_w (-1)^{sw^t} (-1)^{\hat{f}(w)} |w\rangle$$

5. Compute the function $|w\rangle \mapsto (-1)^{\tilde{f}(w)} |w\rangle$ into the phase resulting in,

$$\frac{1}{2^{n/2}} \sum_w (-1)^{sw^t} |w\rangle$$

   where we have used the fact that f is a bent function
6. Finally apply another $H^{\otimes n}$ to get the state $|s\rangle$ and measure s

From this description it is clear that we needed one query to g and one query to $\tilde{f}$ to solve the problem, that the algorithm is exact, and that the overall running time is given by $O(n)$ quantum operations. □

The feature map is similar to a quantum algorithm for estimating the hidden shift in boolean bent functions, as above. The circuit $\mathcal{U}_\Phi$ we consider is indeed very similar to the one discussed in [35]. Whereas the diagonal layers are equivalent in this algorithm, consider if we allow them to differ: $\tilde{\mathcal{U}_\Phi} = U_{\Phi_1(x)} H^{\otimes n} U_{\Phi_2(x)} H^{\otimes n}$.

In particular, the phase gate layers so that U_{Φ_1} and U_{Φ_2} encode a shifted bent-function and the functions dual, then the circuit $\mathcal{A}_1$ is given by $H^{\otimes n} \tilde{\mathcal{U}_\Phi}$. If the interaction graph G is bi-partite, it is possible to encode Maiorana-McFarland bent-functions for $d = 2$ by choosing the corresponding $\varphi_{k,l}(x), \varphi_k(x)$ to be either π or 0. In [35], the diagonal layer in $\mathcal{A}_1$, are queries to an oracle encoding the the shifted bent-function and its dual. It can be shown that with respect to this oracle there is an exponential separation in query complexity over classical query algorithms.

If this conjecture is proven, then an important step is taken for quantum machine learning.

4.2 Quantum Variational Classification

Quantum circuit learning is a hybrid quantum-classical method that allows one to "train" a quantum circuit by classically parameterizing some of its gates. Then, we may train these parameters classically to achieve a defined goal.

Suppose we have an n qubit circuit and labelled data $\{(\vec{x}_i, y_i)\}, \vec{x}_i \in \mathbb{R}^n, y_i \in \{0, 1\}$ and consider the following procedure

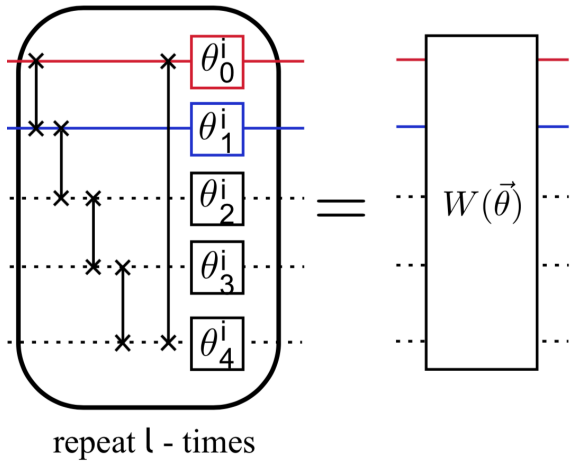


Figure 4.3: Variational circuit diagrammed in [22]

1. Encode classical input data $\{\vec{x}_i\}$ as $\{|\Phi(x_i)\rangle\}$ $(\equiv \{\Phi(x_i)\,|0\rangle\})$ for some unitary feature map φ

2. Apply a θ-parameterized unitary $W(\theta)$ to the input state giving

$$W(\theta)\,|\Phi(x_i)\rangle$$

3. Measure multiple shots of some observables $\{M\}$. The expectation value of these observables is utilized to output a real-valued prediction function $f : \{x_i\} \to \mathbb{R}$. Furthermore $sgn(f(x_i))$ indicates the learned label $\hat{y}_i$ which approximates y_i.

4. Minimize a chosen cost function with is a function of $f(x_i)$. For example, the ℓ_2 loss

$$\sum_i (y_i - f(x_i))^2$$

5. Evaluate the performance and update θ accordingly (e.g. using gradient descent). Return to step 2, or exit if done iterating.

A common choice of $W(\theta)$ (Figure 4.2) is the following circuit: Let $W(\theta)$ have l layers i.e. $W(\theta) = W_1(\theta)\cdots W_l(\theta)$.

Now, for each $W_i(\theta)$ we construct the following circuit: label the qubits $\{1, \cdots, n\}$ and consider the nearest-neighbor pairing $N = \{(1,2),(2,3),\cdots,(n-1,n)\}$. Then, perform Control-Z on each pair in N. Next, write $\theta = (\theta_1, \cdots, \theta_n)$ where each θ_i is a vector specified by Euler angles. Hence, we can perform rotation $R(\theta_i)$ on the ith qubit.

4.3 Experimental Simulation of the Shifted Bent Feature Map

4.3.1 Separable Data

Here, we demonstrate an experimental construction of the feature map defined in (4.2) considering 2 qubits for data input. We use the Strawberry Fields package for simulation [25].

We generate data so that it can be perfectly separated. Choose coefficients

$$\varphi_i(x) = x_i, i \in \{1, 2\}$$
$$\varphi_{\{1,2\}}(x) = (\pi - x_1)(\pi - x_2)$$

and draw a fixed random element $V \in SU(4)$. Define $\Pi = Z_1 Z_2$ to be parity on 2 qubits. Then, generate data by drawing (uniform) random $x \in (0, 2\pi]^2$ and labelling based on decision rule

$$f(x) = \langle 0 | \mathcal{U}^\dagger_{\Phi(x)} V^\dagger \Pi V \mathcal{U}_{\Phi(x)} | 0 \rangle$$

In particular, define a threshold Δ and if $|f(x)| \leq \Delta$ then draw another x. Otherwise, label by $sgn(f(x))$. We draw 40 such x and arbitrary partition the dataset into training and test subsets of size 20 each.

We then use the variational circuit described in the previous section to train our classifier. In particular, the variational circuit trains some $W(\theta)$ parameterized by θ and then constructs decision rule

$$\tilde{f}(x) = \langle 0 | \mathcal{U}^\dagger_{\Phi(x)} W(\theta)^\dagger \Pi W(\theta) \mathcal{U}_{\Phi(x)} | 0 \rangle)$$

and classifies by $sgn(\tilde{f}(x))$. In principle, then, because $W(\theta)$ can learn any $SU(4)$ gate for appropriately chosen θ, we should be able to classify to near perfect accuracy. We use the "Nesterov momentum optimizer" to train θ, an improvement of the usual stochastic gradient descent.

Below, we plot the results of a few representative sample trials exploring various values of Δ.

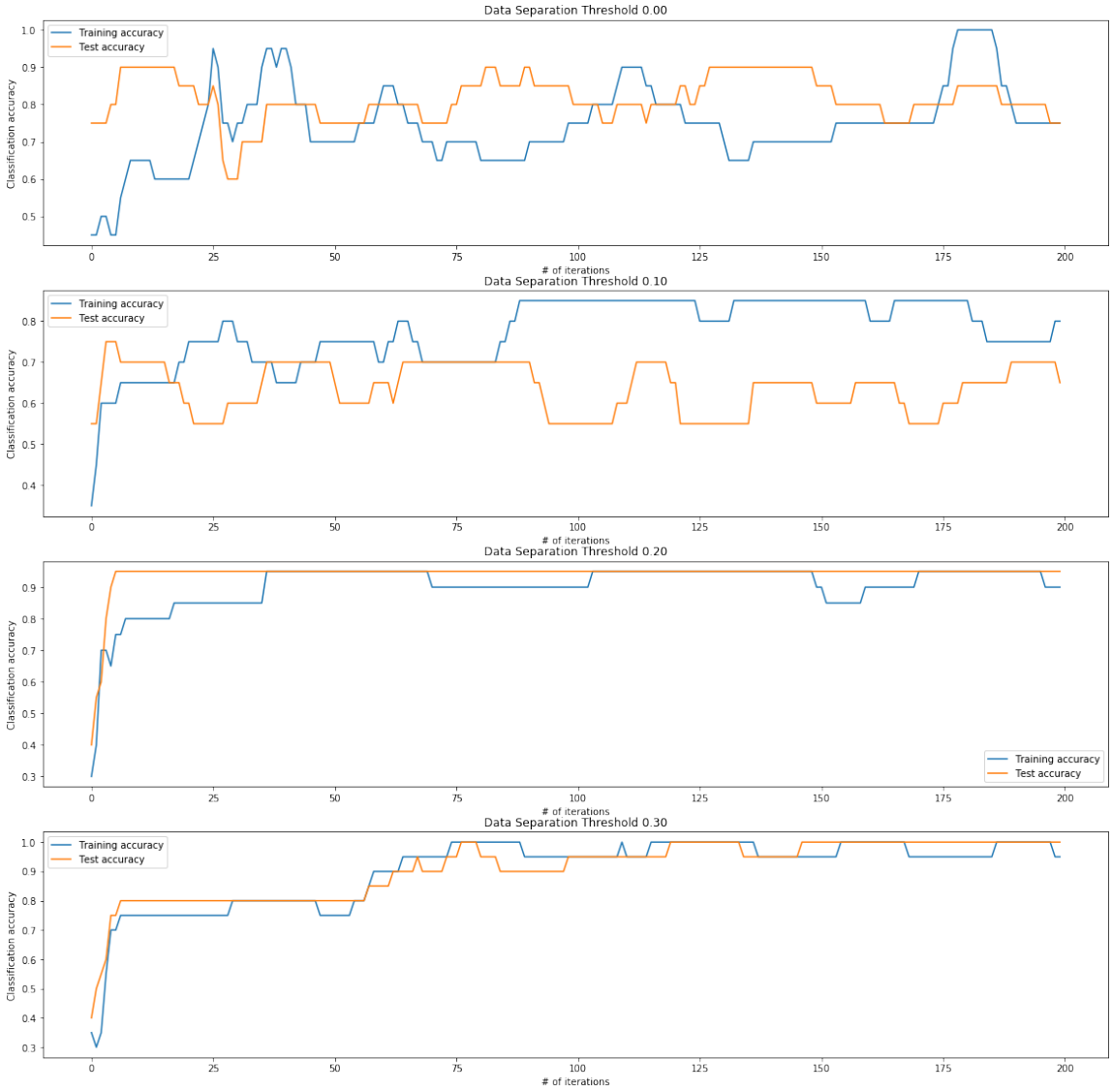


Figure 4.4: A comparison of training and test convergence across varying data separations.

These results show that we converge to near-perfect classification for all of these positive separations. However, the greater the separation, the faster the convergence. We can interpret this as reflecting the fact that we are less likely to overfit or end up in a suboptimal equilibrium as part of gradient descent process if the data is well-separated.

Furthermore, can view the convergence of the dual ℓ^2 loss view below.

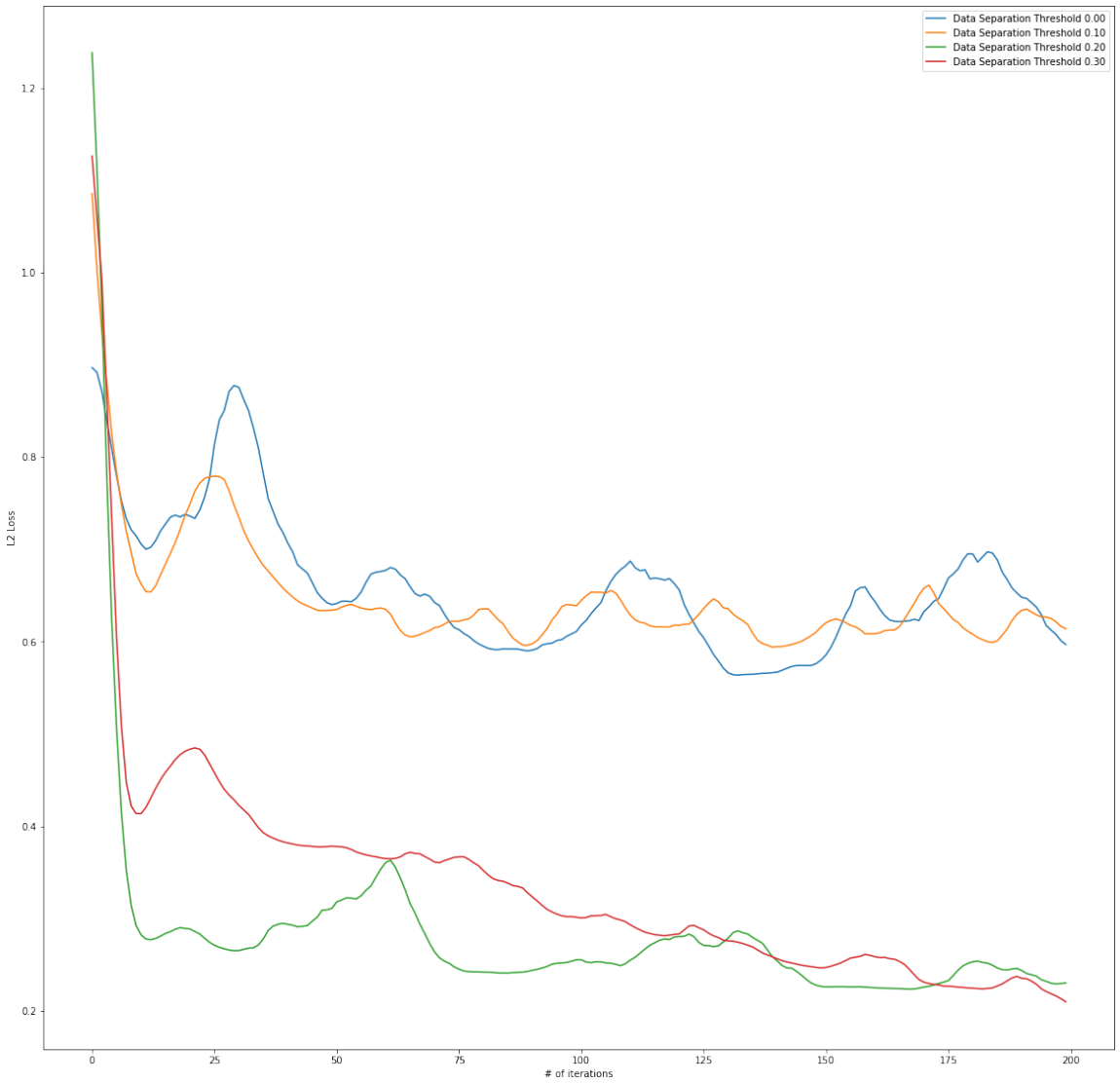


Figure 4.5: A similar comparison of ℓ^2 loss convergence across varying data separations.

Visually, generated data may look as below for some $V \in SU(4)$. We can visualize the decision boundary of the kernel by shading the regions in a color indicating the label which the classifier would output at each point.

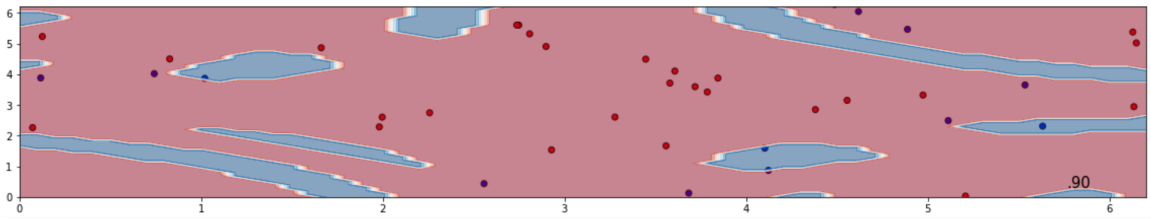


Figure 4.6: Sample output of the classifier. The colors of the regions indicate the label for which the classifier has learned for each point. The circles indicate data points in the generated set, colored by true label.

One important question we wished to resolve is the relationship between the convergence in training accuracy and the number of layers utilized by the variational circuit. Hence, we checked convergence for two separable datasets across the range of 1-5 layers, performing 20 trials per layer depth. Based on the preceding results, data separation of 0.3 was utilized in order to speed up convergence.

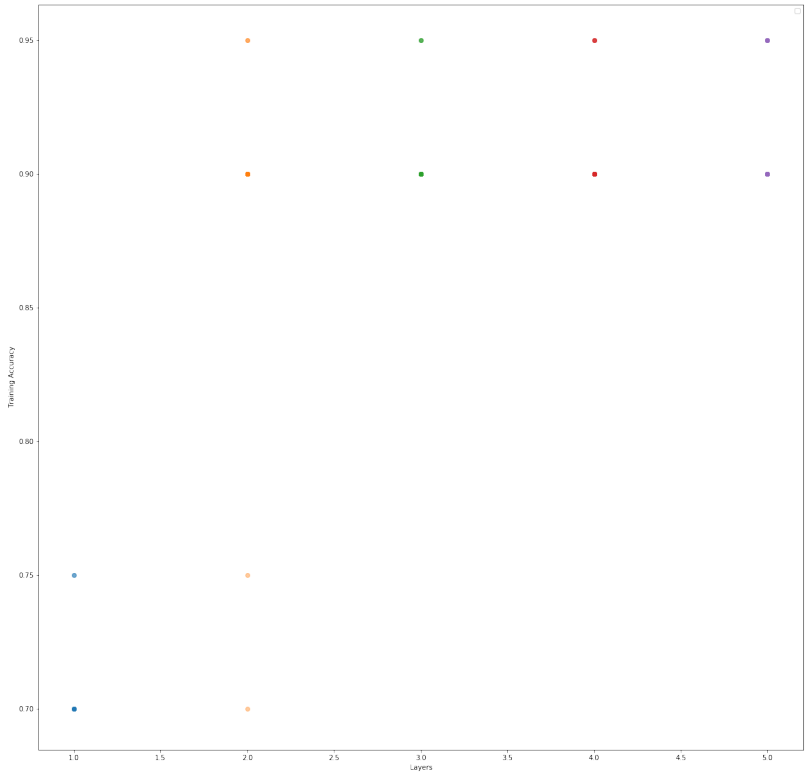


Figure 4.7: Training accuracy over repeated trials of 200 epochs varying over layer depth

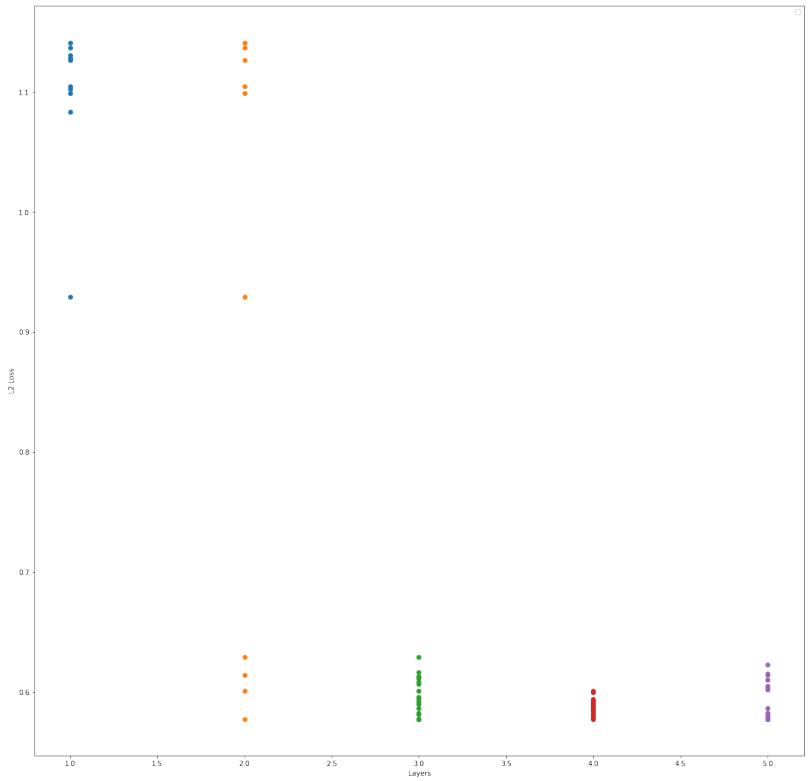
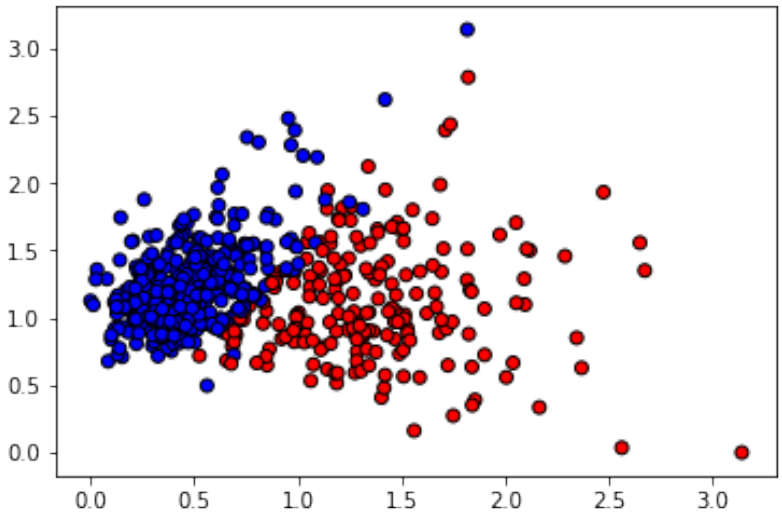


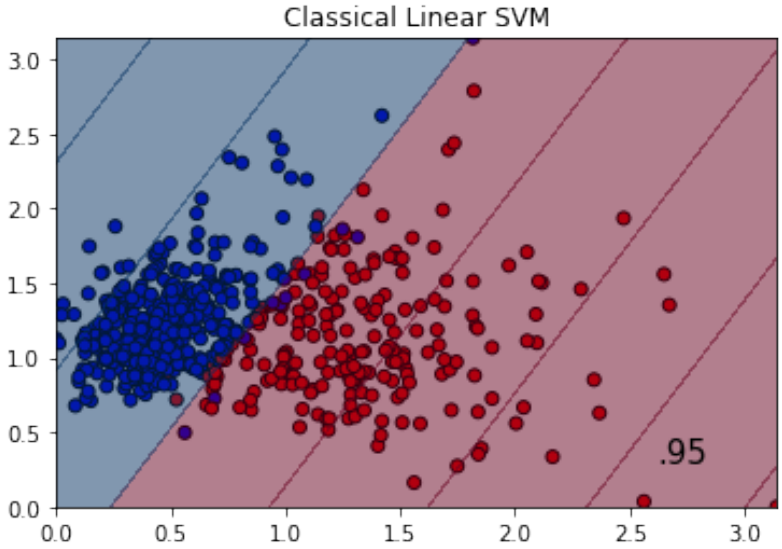
Figure 4.8: ℓ^2 loss over repeated trials of 200 epochs varying over layer depth

Here, we can see that 3 layers is sufficient to regularly achieve convergence of the classifier to $> 90\%$ training accuracy.

4.3.2 Practical Data

Furthermore, we contribute an analysis of the performance of the quantum variational to practical data. In particular, consider the UCI ML Breast Cancer Dataset which contains 30 features and 569 samples. The labels correspond to whether a cell mass is malignant or benign. We apply PCA to reduce the dimensionality of the dataset to 2 and standardize to a centered Gaussian about $[0, \pi]$.





Here, we see that the data is roughly linearly separable. In fact, a linear classifier, such as SVM with the standard Euclidean kernel map, is able to classify this dataset with 0.95 accuracy. Hence, neither the kernel trick nor regularization are necessary to perform well. Nevertheless, it is instructive to check that the quantum bent kernel with a variational circuit can achieve reasonable accuracy.

Hence, below we provide an analogous plot to the previous subsection showing the range of training accuracies and ℓ^2 loss outcomes for varying variational circuit depths.

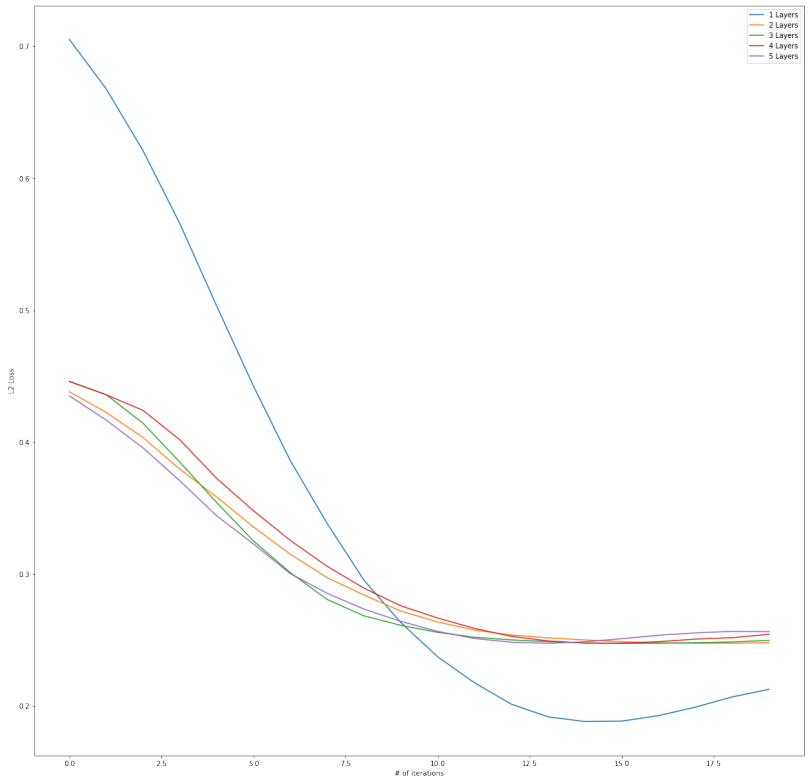


Figure 4.9: The pattern in convergence in cost on the UCI Breast Cancer dataset is similar to the one for the separable dataset.

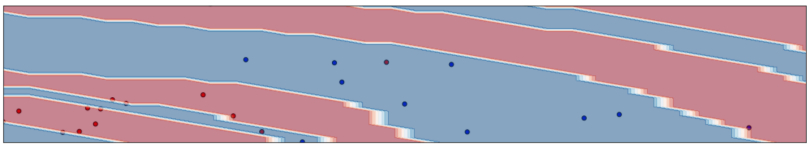


Figure 4.10: Sample output of the classifier for the UCI Breast Cancer dataset

4.3.3 Moons Data

We generate the data of this section by using the scikit-learn Python package and add random Gaussian noise to the $\{x_i\}$ for each trial.

The data appears similarly to below and one can see that the data is no longer nearly linearly separable. Hence, it is certainly appropriate to consider the kernel trick in this context.

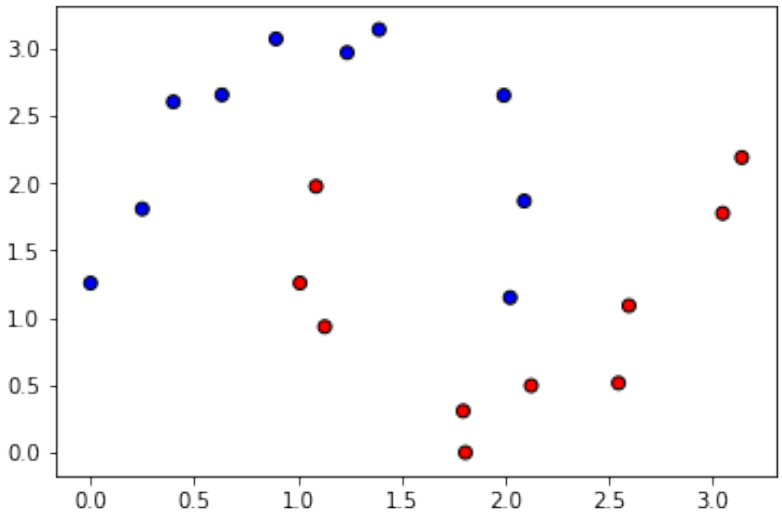


Figure 4.11: Sample Moons data for some perturbation in Gaussian noise

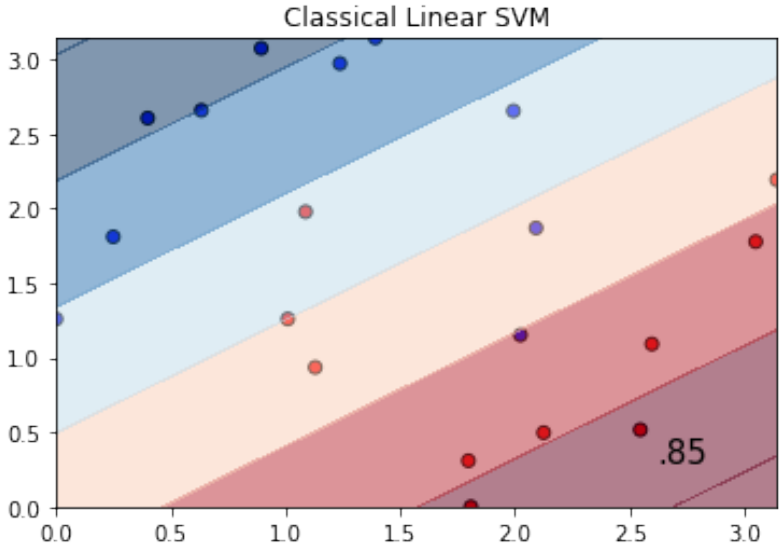


Figure 4.12: Performance of SVM with a linear kernel on the Moons dataset

To see how the quantum feature map performs, we direct your attention to the next section.

4.4 Theoretical Extensions of the Shifted Bent Feature Map

4.4.1 A Connection Between State Tomography and Hyper-parameter Tuning

One method to improve the performance of a kernel map is to tune some set of hyper-parameters which adjust the embedding in the higher-dimensional feature space. Hence, we consider introducing a set of hyper-parameters which tunes the extent to which nonlinearity enters the phase of the $U_{\Phi(x)}$ (Equation 4.1). In particular, consider a d-dimensional classification problem. Hence, the phase of Equation 4.1 can be decomposed as d summations which increase in the subset size of S. For each the $d-1$ terms where $|S| \geq 2$, we may add an γ_i real coefficient. Hence, this gives us a revised diagonal (in the Z basis) unitary within the feature map (Equation 4.2):

$$U_{\Phi(x;\vec{\gamma})} = \exp\left(i\Big[\sum_i \varphi_i(x) Z_i + \gamma_1 \sum_{i,j} \varphi_{i,j}(x) Z_i Z_j + \cdots + \gamma_{d-1} \sum_{S\subseteq[d],|S|=d} \varphi_S(x) \prod_{i\in S} Z_i\Big]\right)$$

Figure 4.4.1 offers a representative view of how adjusting the single γ hyper-parameter for the 2-qubit map adjusts the classification circuit to which the variational algorithm converges after 200 epochs when considering the UCI Breast Cancer dataset.

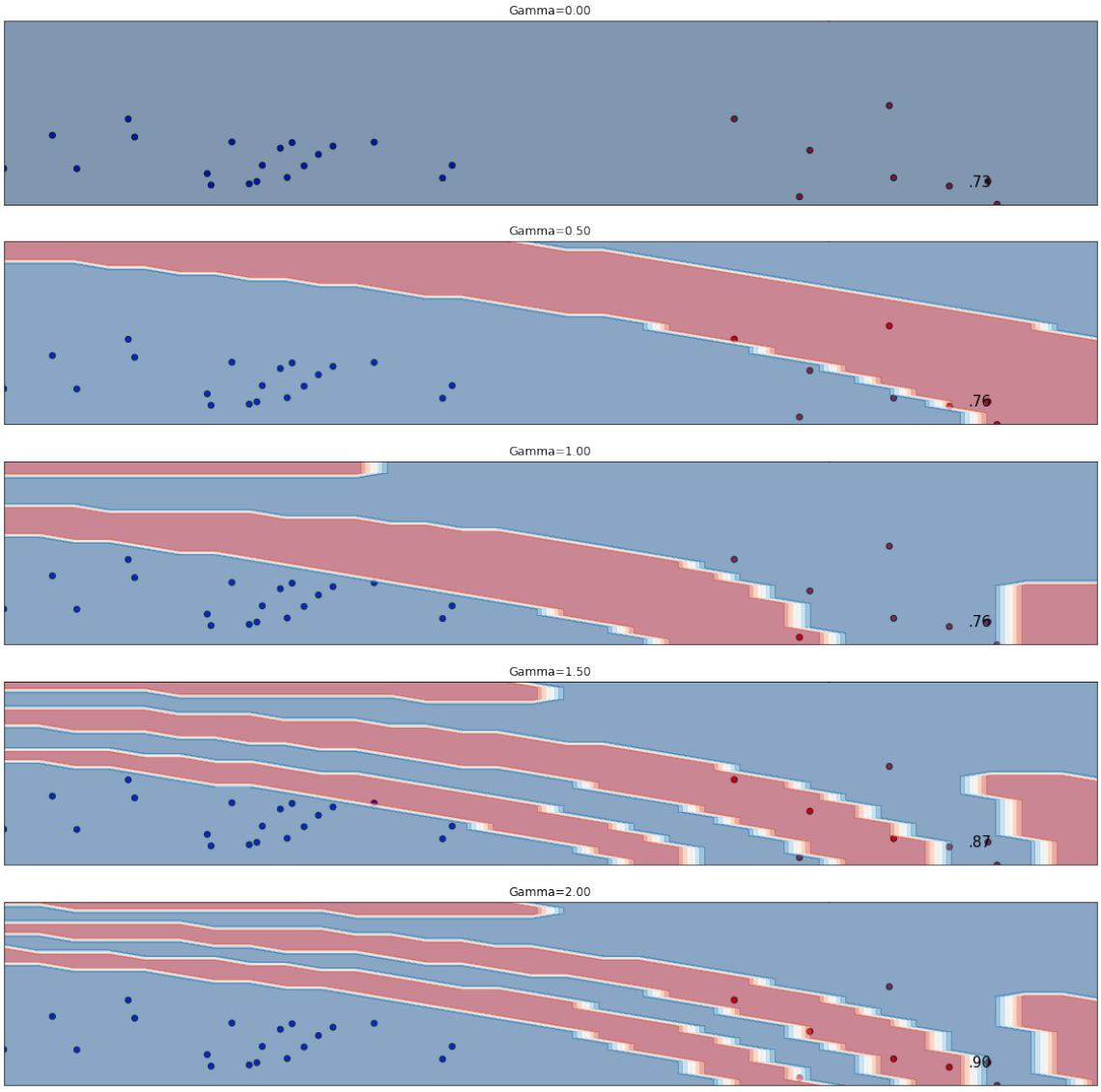


Figure 4.13: Performance of the variational classification circuit (for 200 epochs) on a random sample of the UCI Breast Cancer dataset for varying 2-point nonlinearity parameter γ. The classification accuracy (bottom-right corner) is seen to increase as γ increases for this trial. Note that the number of $+1$ vs. -1 data points is unbalanced in this case.

We contribute a novel method of thinking about tuning hyper-parameters for a quantum feature map and additionally demonstrate that the method works well in practice by returning to the "moons" binary classification problem of Section 4.3.3.

First, we make the straightforward observation that the goal of the variational classifier can be restated as solving a state identification problem:

Write that a data point x is encoded as

$$|\Phi(x;\vec{\gamma})\rangle \equiv \mathcal{U}_{\Phi(x;\vec{\gamma})}\,|0\rangle$$

Then, consider the density matrices

$$\rho_+ = \frac{1}{N_+}\sum_{x_i:y_i=+1} |\Phi(x_i;\vec{\gamma})\rangle\,\langle\Phi(x_i;\vec{\gamma})|$$
$$\rho_- = \frac{1}{N_-}\sum_{x_i:y_i=-1} |\Phi(x_i;\vec{\gamma})\rangle\,\langle\Phi(x_i;\vec{\gamma})|$$

Therefore, the goal of binary classification can be restated as finding the optimal pair of POVMs so as to maximize the probability of correctly distinguishing these two density matrices. As we've discussed in Section A.7.2, the trace distance provides an achievable (by the Helstrom measurement) upper bound for solving this state identification problem. Hence, our method of hyperparameter tuning can be stated as follows: (1) compute the density matrices ρ_+, ρ_- above for a set of candidate $\{\vec{\gamma_i}\}$ (2) computing the trace distance $T(\rho_+, \rho_-)$ across the set of $\{\vec{\gamma_i}\}$, and (3) choose the $\vec{\gamma_i}$ which maximizes trace distance.

For the dataset of Figure 4.4.1, we indeed see that trace distance increases with gamma within the range $[0.0, 2.0]$.

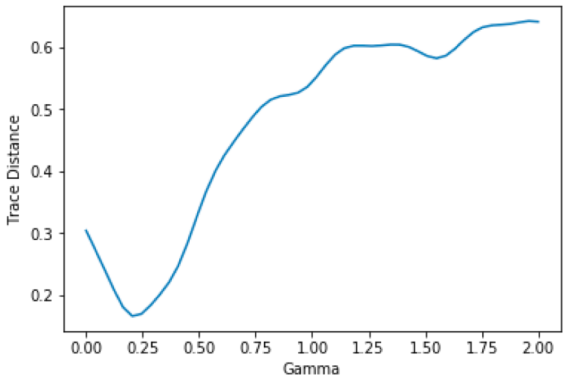


Figure 4.14: Trace distance as a function of γ for a random sample of the UCI Breast Cancer dataset

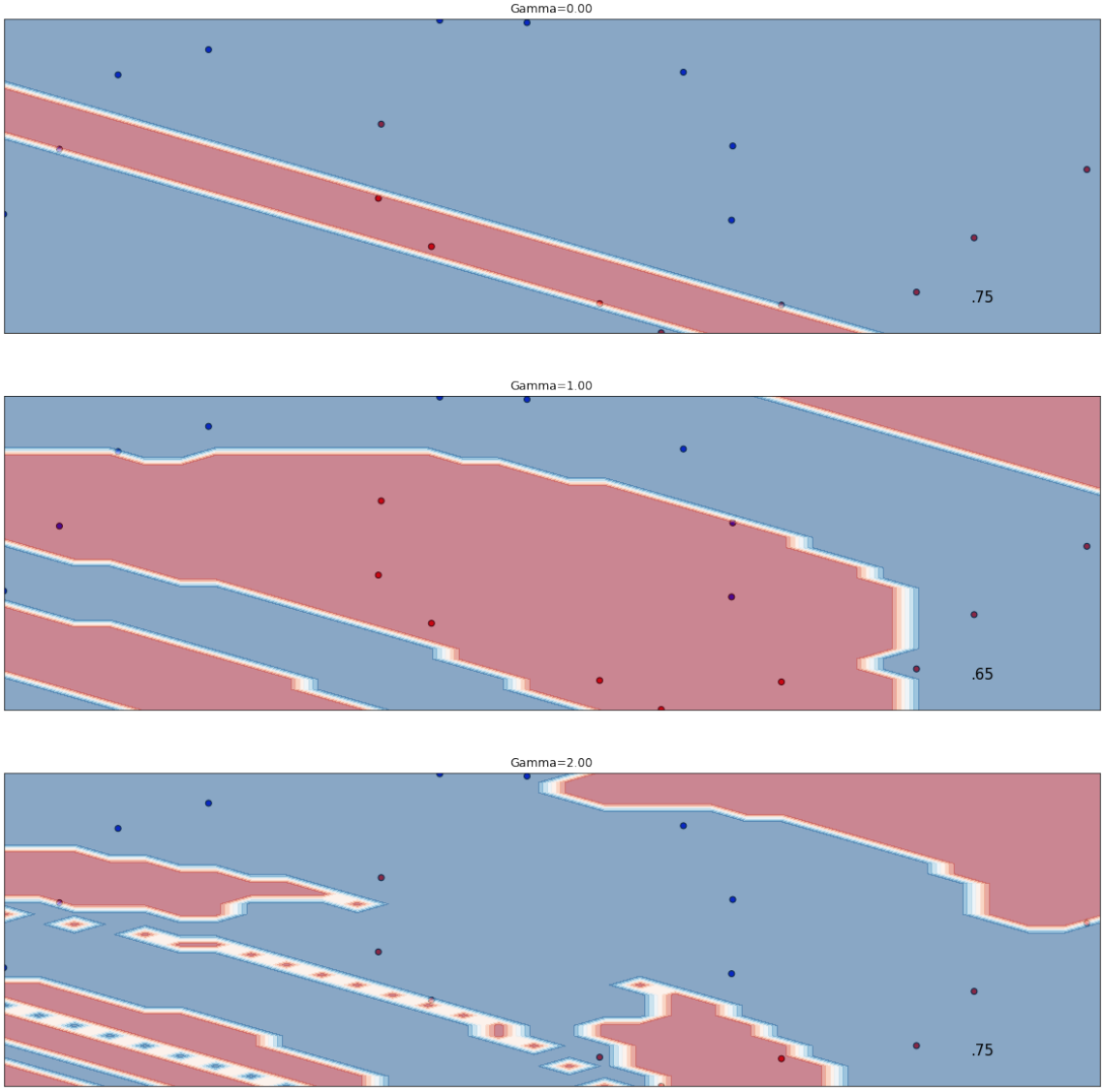
Do note that the trace distance can be computed efficiently on a quantum computer using the method of Quantum Phase Estimation A.5.2 with Hamiltonian $e^{-i(\rho_+ - \rho_-)t}$. In the following section, we will extensively evaluate the classification performance of this method.

4.4.2 Empirical Evaluation of Tuning by Trace Distance

To see that the hypothesized relationship between trace distance and the optimal choice of $\vec{\gamma}$ holds for the roughly linearly separable practical data of the previous section, we performed 20 trials of 200 epochs for linearly spaced γ on two sets of 20 random samples of the practical dataset,

now requiring that the ratio between $+1$ and -1 labelled data points is balanced. As a result for $\gamma = \{0.0, 0.25, 0.5, 1.0, 1.5, 2.0\}$ we found that the optimal training accuracy was given by $\{0.55, 0.55, 0.60, 0.80, 0.85, 0.85\}$.

Now, we consider the Moons dataset over varying γ



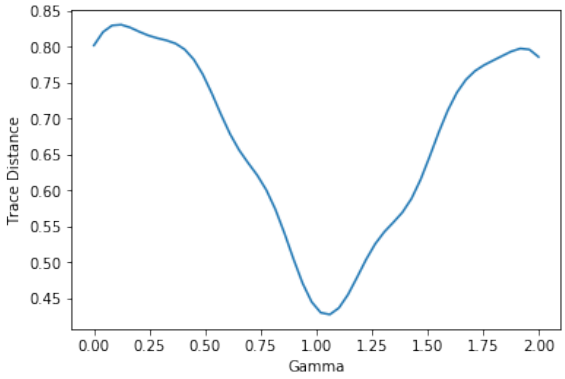
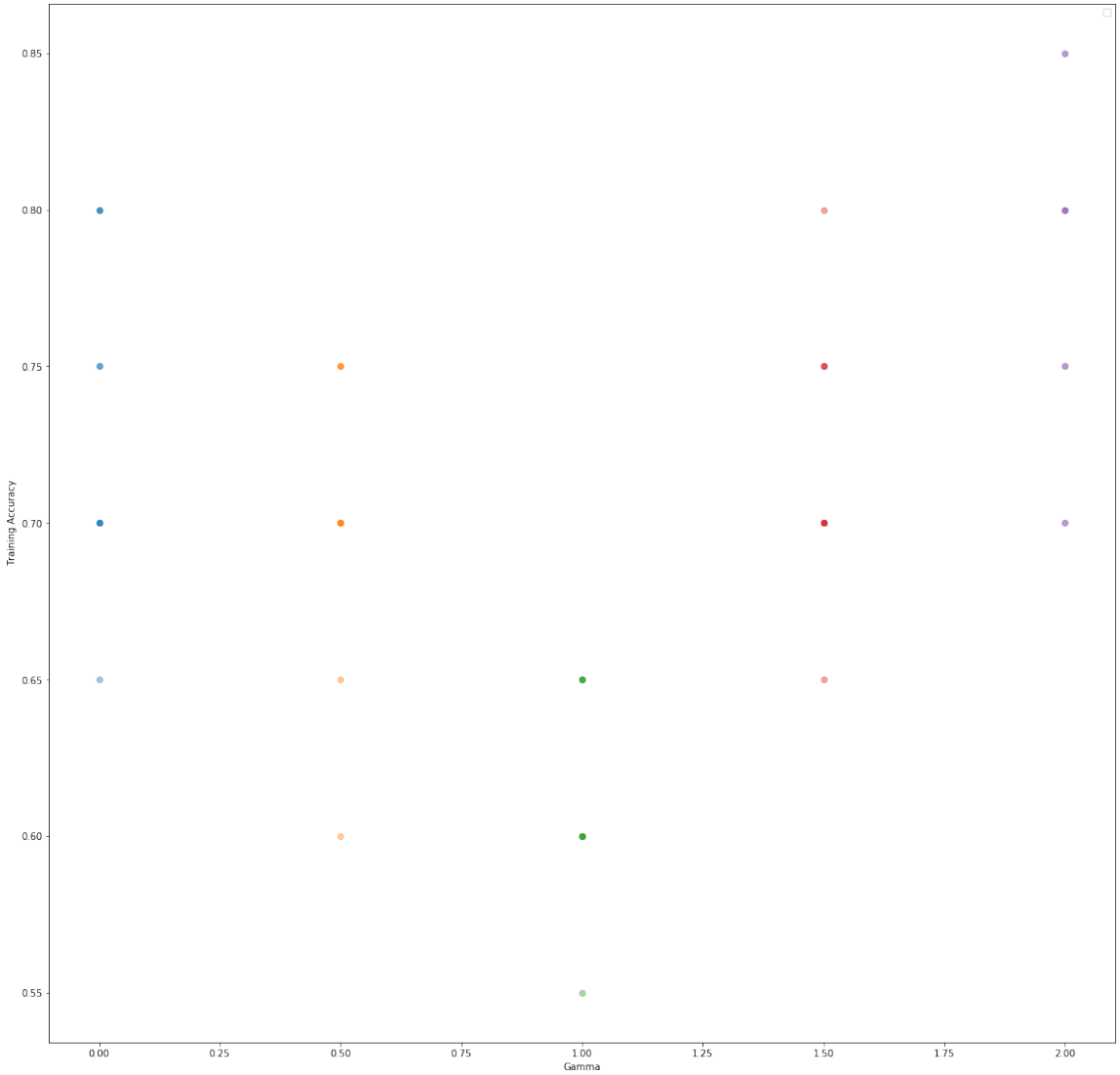


Figure 4.15: The structure of the moons dataset empirically requires a V shaped curve with the trough roughly near $\gamma = 1.0$. Note that such a γ is the default value used with no hyper-parameter tuning.

From the figures above, we expect the classifier to perform relatively poorly for $\gamma = 1.0$ and increase towards the extremities. Again, we performed 20 trials of 200 epochs for linearly spaced γ on two sets of 20 random samples of the Moons dataset with noise, now requiring that the ratio between $+1$ and -1 labelled data points is balanced. The figure below captures our results.



This clearly matches with the expected results. Hence, we hope that future works can extend these results and identify additional datasets that have a particular structure for which γ ought to be tuned and leave it to these works to verify that trace distance serves a useful purpose to this end.

Chapter 5

Quantum-Inspired Length-Square Sampling

As we've seen, most well-known QML algorithms convert input quantum states to a desired output state or value. Thus, they do not provide a routine to get necessary copies of these input states (a state preparation routine) and a strategy to extract information from an output state. Both are essential to making the algorithm useful.

We've also seen that that many of the initial "practical" quantum algorithms attempted to workaround this difficulty by considering low-rank linear algebraic problems (poly-logarithmic in matrix dimension). For example, the data fitting algorithm [42] allows one to compute $A^+ |b\rangle = |x_{LS}\rangle$ in $\tilde{O}(log(N)(s^3\kappa^6)/\epsilon)$ time (query complexity) by using the HHL algorithm and matrix multiplication. Hence, we require that A is sparse with low condition number κ as well.

Nevertheless, these low-rank quantum algorithms still have substantial help from an assumed input state preparation routine which has been shown to be at least $\Omega(\sqrt{n})$ in vector input size ([40]). Hence, a natural question to ask is whether there's a similar classical data structure which takes time polynomial in quantum state preparation which can offer machine learning algorithms with similar guarantees. In particular, we compare quantum algorithms with quantum state preparation to classical algorithms with sample and query access to input.

5.1 Definitions

We remain consistent with the definitions of Section 2.3.2.

5.2 Low-Rank Estimation

Low-rank estimation is given by the FKV algorithm detailed in Section 2.3.2.

5.3 Trace Inner Product Estimation

The first tool is a classical analog to the quantum SWAP test, which computes the inner product of two pure states. For $x, y \in \mathbb{C}^n$, if we are given that $x \in \mathcal{SQ}$ and $y \in \mathcal{Q}$, then we can estimate $\langle x, y\rangle$

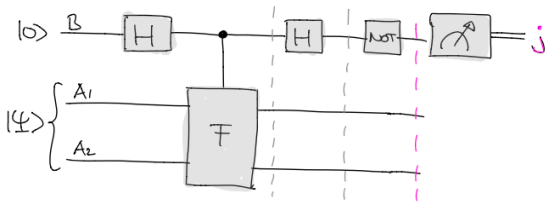


Figure 5.1: Quantum SWAP test (from [41])

with probability $\geq 1-\delta$ and error $\epsilon\|x\|\|y\|$

Fact 5.3.1. *For $\{X_{i,j}\}$ i.i.d random variables with mean μ and variance σ^2, let*

$$Y := \underset{j\in[\log 1/\delta]}{\text{median}} \underset{i\in[1/\epsilon^2]}{\text{mean}} X_{i,j}$$

Then $|Y-\mu| \leq \epsilon\sigma$ with probability $\geq 1-\delta$, using only $O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$ samples.

In words, we may create a mean estimator from $1/\epsilon^2$ samples of X. Hence, we can then compute the median of $\log 1/\delta$ such estimators. Naturally, one may ask why we do not simply take the empirical mean. Catoni (2012) shows that Chebyshev's inequality is the best guarantee one can provide when considering pure empirical mean estimators for an unknown distribution (and finite mean and variance). Hence, "median of means" provides an exponential improvement in probability of success $(1-\delta)$ guarantee

Corollary 5.3.1.1. *For $x, y \in \mathbb{C}^n$, given $x \in \mathcal{SQ}$ and $y \in \mathcal{Q}$, we can estimate $\langle x, y\rangle$ to $\epsilon\|x\|\|y\|$ error with probability $\geq 1-\delta$ with query complexity $O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$*

Proof. Sample an **index** s from x. Then, define $Z := x_s y_s \frac{\|y\|^2}{|y_s|^2}$. Apply the Fact with $X_{i,j}$ being independent samples Z. □

5.4 Least-Square Sample Generation

For $V \in \mathbb{C}^{n\times k}, w \in \mathbb{C}^k$, given $V^\dagger \in \mathcal{SQ}$ (*column*-wise sampling of V) and $w \in \mathcal{Q}$, we can simulate $Vw \in \mathcal{SQ}$ with poly(k) queries. In words, if we can least-square sample the columns of matrix V and query the entries of vector w, then:

1. We can query entries of their multiplication (Vw)
2. We can least-square sample from a distribution that emulates their multiplication

Hence, as long as $k \ll n$, we can perform each using a number of steps polynomial in the number of columns of V.

Algorithm 5.4.1. *(Rejection sampling)*
Input: Samples from distribution P
Output: Samples from distribution Q
The procedure is as follows:

- *Sample s from P*
- *Compute $r_s = \frac{1}{N}\frac{Q(s)}{P(s)}$, for fixed constant N*
- *Output s with probability r_s and restart otherwise*

Fact 5.4.2. *If $r_i \leq 1, \forall i$, then the above procedure is well-defined and outputs a sample from Q in N iterations in expectation.*

Proposition 5.4.3. *For $V \in \mathbb{R}^{n\times k}$ and $w \in \mathbb{R}^k$, given $V^\dagger \in \mathcal{SQ}$ and $w \in \mathcal{Q}$, we can simulate $Vw \in \mathcal{SQ}$ with expected query complexity $\tilde{O}((\frac{1}{\epsilon^2}\log\frac{1}{\delta}))$*

Proof. Clearly, we can compute entries $(Vw)_i$ with $O(k)$ queries.

Furthermore, we can sample using rejection sampling: Let P be the distribution formed by sampling from $V_{(\cdot,j)}$ and Q is the target Vw. Hence, compute r_s to be a constant factor of Q/P

$$r_i = \frac{\|w^T V_{\cdot,i}\|^2}{\|w\|^2\|V_{\cdot,i}\|^2}$$

Notice that we can compute these r_i's (in fact, despite that we cannot compute probabilities from the target distribution), and that the rejection sampling guarantee is satisfied (via Cauchy-Schwarz). Since the probability of success is $\|Vw\|^2/\|w\|^2$, it suffices to estimate the probability of success of this rejection sampling process to estimate this norm. Through a Chernoff bound, we see that the average of $O(\|w\|^2(\frac{1}{\epsilon^2}\log\frac{1}{\delta}))$ "coin flips" is in $[(1-\epsilon)\|Vw\|, (1+\epsilon)\|Vw\|]$ with probability $\geq 1-\delta$. □

5.5 Application: Stochastic Regression

For a low-rank matrix $A \in \mathbb{R}^{m\times n}$ and a vector $b \in \mathbb{R}^n$, given $b, A \in \mathcal{SQ}$, (approximately) simulate $A^+b \in \mathcal{SQ}$.

Algorithm 5.5.1. *The procedure is as follows:*

- *Low-rank approximation (3) gives us $S, \hat{U} \in \mathcal{SQ}$.*
- *Applying thin-matrix vector (2), we get $\hat{V} \in \mathcal{SQ}$, where $\hat{V} := S^T\hat{U}$; we can show that the columns of $\hat{V}$ behave like the right singular vectors of A.*
- *Let $\hat{U}$ have columns $\{\hat{u}_i\}$. Hence, $\hat{V}$ has columns $\{S\hat{u}_i\}$. Write its ith column as $\hat{v}_i := S\hat{u}_i$.*
- *Low-rank approximation (3) also outputs the approximate singular values $\hat{\sigma}_i$ of A*

Now, we can write the approximate vector we wish to sample in terms of these approximations:

$$A^+b = (A^TA)^+A^Tb \approx \sum_{i=1}^{k}\frac{1}{\hat{\sigma}_i^2}\hat{v}_i\hat{v}_i^TA^Tb$$

- We approximate $\hat{v}_i^TA^Tb$ to additive error for all by noticing that $\hat{v}_i^TA^Tb = \mathrm{tr}\big(A^Tb\hat{v}_i^T\big)$ is an inner product of A^T and $b\hat{v}_i^T$.

- Thus, we can apply (1), since being given $A \in \mathcal{SQ}$ implies $A^T \in \mathcal{SQ}$ for A^T viewed as a long vector.
- Define the approximation of $\hat{v}_i^T A^T b$ to be $\hat{\lambda}_i$. At this point we have (recalling that $\hat{v}_i := S\hat{u}_i$)

$$A^+ b \approx \sum_{i=1}^{k} \frac{1}{\hat{\sigma}_i^2} \hat{v}_i \hat{\lambda}_i = S \sum_{i=1}^{k} \frac{1}{\hat{\sigma}_i^2} \hat{u}_i \hat{\lambda}_i$$

- Finally, using (2) to provide sample access to each $S\hat{u}_i$, we are done! $\tilde{O}(\kappa^{16} k^6 \|A\|_F^6 / \epsilon^6)$ complexity.

5.5.1 Definitions and Assumptions

Let $b \in \mathbb{C}^m$ and $A \in \mathbb{C}^{m \times n}$ s.t. $\|A\| \leq 1$ where $\|\cdot\|$ signifies the operator norm (or spectral norm). Furthermore, require that $\text{rank}(A) = k$ and $\|A^+\| \leq \kappa$ where A^+ is the pseudoinverse of A. Hence, observe that $\|A\| \leq 1$ is equivalent to A having maximum singular value 1[1]. Similarly, A^+ has inverted singular values from A and so $\|A^+\|$ is equal to the reciprocal of the minimum nonzero singular value. Therefore, the condition number of A is given by $\|A\|\|A^+\| \leq \kappa$.

So, define x to be the least-squares solution to the linear system $Ax = b$ i.e. $x = A^+ b$. Then, in terms of these definitions, we define two primary goals:

1. Query a vector $\tilde{x}$ s.t. $\|\tilde{x} - x\| \leq \epsilon \|x\|$
2. Sample from a distribution that approximates $\frac{|x_j|^2}{\|x\|^2}$ within total variation distance (Theorem 2.3.1) 2ϵ.

In order to do this, we simply assume that we have length-square sampling access to A. In other words, we are able to sample row indices of A from the distribution $\frac{\|A_{(i,\cdot)}\|^2}{\|A\|_F^2}$

5.5.2 Sequence of Approximations

First, we'll summarize the sequence of approximations that we'll perform using length-squared sampling techniques. We'll describe these steps in depth in the following sections.

Of course, we know that the least squares solution of the linear system is given by the orthogonal projection

$$(A^\dagger A)^+ A^\dagger = A^+ b$$

So, we first approximate $A^\dagger A$ by $R^\dagger R$ where $R \in \mathbb{C}^{r \times n}$, $r \ll m$ is constructed from length-square sampling r rows of A. Now, denote the spectral decomposition

$$A^\dagger A \approx R^\dagger R = \sum_{l=1}^{k} \frac{1}{\sigma_l^2} \left| v^{(l)} \right\rangle \left\langle v^{(l)} \right|$$

[1] To see this, simply consider Spectral Theorem applied to Hermitian matrix $A^\dagger A$

where of course σ_i and $\left|v^{(i)}\right\rangle \in \mathbb{C}^n$ are the singular values and right singular vectors of R, respectively.

We see that computing these right singular vectors of R can still be computationally prohibitive given the dimension n. Hence, we can use length-square sampling again, this time on the columns of R to give a matrix $C \in \mathbb{C}^{r\times c}$, $c \ll n$. Now, the left singular vectors of C which we denote as $\left|w^{(i)}\right\rangle \in \mathbb{C}^r$ can be efficiently computed via standard SVD methods. So,

$$RR^\dagger \approx CC^\dagger = \sum_{l=1}^{k} \frac{1}{\sigma_l^2} \left|w^{(l)}\right\rangle \left\langle w^{(l)}\right|$$

We can then show that ()

$$\left|\tilde{v}^{(i)}\right\rangle := R^\dagger \left|w^{(l)}\right\rangle / \tilde{\sigma}_l \tag{5.1}$$

provides a good approximation of $\left|v^{(i)}\right\rangle$. Note that $\tilde{\sigma}_l$ are the singular values of C which then approximate the singular values of R which similarly approximate the singular values of A. This follows from $A^\dagger A \approx R^\dagger R$ and $RR^\dagger \approx CC^\dagger$ by the Hoffman–Wielandt inequality detailed in Lemma 2.7 of [23] and stated without proof below.

Lemma 5.5.2. *Hoffman–Wielandt inequality*
If P, Q are two real, symmetric $n\times n$ matrices and $\lambda_1, \cdots \lambda_n$ denote eigenvalues in non-decreasing order, then

$$\sum_{t=1}^{n} (\lambda_t(P) - \lambda_t(Q))^2 \leq \|P - Q\|_F^2$$

At this point, it seems like we haven't made much progress since computing $R^\dagger \left|w^{(l)}\right\rangle$ is still expensive. However, it turns out that all we need to enable query access to $\tilde{x}$ is the ability to efficiently estimate the trace inner product $\mathrm{tr}\left(U^\dagger V\right)$ where U and V are operators such that U can be the length-square sampled and V can be queried. To see this, we write our solution, $\tilde{x}$, in terms of the approximations thus far

$$\begin{aligned}
\tilde{x} &\approx A^+ \left|b\right\rangle \\
&\approx (R^\dagger R)^+ A^\dagger \left|b\right\rangle \\
&\approx \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^2} \left|\tilde{v}^{(l)}\right\rangle \left\langle \tilde{v}^{(l)}\right| A^\dagger \left|b\right\rangle
\end{aligned}$$

Hence, define $U := A$, $V := \left|b\right\rangle \left\langle \tilde{v}^{(l)}\right|$ in which case

$$\begin{aligned}
\mathrm{tr}\left(U^\dagger V\right) &= \mathrm{tr}\left(A^\dagger \left|b\right\rangle \left\langle \tilde{v}^{(l)}\right|\right) \\
&= \mathrm{tr}\left(\left\langle \tilde{v}^{(l)}\right| A^\dagger \left|b\right\rangle\right) \\
&= \left\langle \tilde{v}^{(l)}\right| A^\dagger \left|b\right\rangle
\end{aligned}$$

since $\langle \tilde{v}^{(l)} | A^\dagger |b\rangle$ is a scalar. Therefore, say that

$$\tilde{\lambda}_l \approx \mathrm{tr}\Big(A^\dagger |b\rangle \left\langle \tilde{v}^{(l)} \right|\Big)$$

and assume that we can compute and memoize these scalars $\tilde{\lambda}_i$ efficiently. In which case,

$$\tilde{x} \approx \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^2} \left|\tilde{v}^{(l)}\right\rangle \tilde{\lambda}_l$$

Recalling the definition of $\left|\tilde{v}^{(i)}\right\rangle$ (5.1),

$$= \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^3} R^\dagger \left|w^{(l)}\right\rangle \tilde{\lambda}_l$$
$$= R^\dagger \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^3} \left|w^{(l)}\right\rangle \tilde{\lambda}_l$$

and so defining $z := \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^3} \left|w^{(l)}\right\rangle \tilde{\lambda}_l$,

$$= R^\dagger z$$

We see that we can compute z efficiently (and memoize it for future queries) because it is a k-linear combination of left singular vectors in $\mathbb{C}^r$. So, say that we wish to query an element $\tilde{x}_j$. We can simply query column $R_{\cdot,j} \in \mathbb{C}^r$ (or equivalently row $R^\dagger_{j,\cdot}$) and compute $R_{\cdot,j} \cdot z$. Hence, we've achieved our first goal.

In order to achieve our second goal, enabling sample access to a distribution that approximates $\frac{|x_j|^2}{\|x\|^2}$, we require one more trick: rejection sampling which we detail in Section ().

All in all, we've performed the chain of approximations,

$$\begin{aligned}
|x\rangle = A^+ |b\rangle &= (A^\dagger A)^+ A^\dagger |b\rangle \\
&\approx (R^\dagger R)^+ A^\dagger |b\rangle = \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^2} \left|v^{(l)}\right\rangle \left\langle v^{(l)}\right| A^\dagger |b\rangle \\
&\approx \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^2} \left|\tilde{v}^{(l)}\right\rangle \left\langle \tilde{v}^{(l)}\right| A^\dagger |b\rangle \\
&\approx \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^2} \left|\tilde{v}^{(l)}\right\rangle \tilde{\lambda}_l = R^\dagger \sum_{l=1}^{k} \frac{1}{\tilde{\sigma}_l^3} \left|w^{(l)}\right\rangle \tilde{\lambda}_l = R^\dagger z
\end{aligned}$$

Now that we've sketched the steps of this process, we detail each approximation and show that we can achieve the claimed correctness and complexity bounds.

5.6 Conclusions

It is Tang's belief that for machine learning problems, $\mathcal{SQ}$ assumptions are more reasonable than state preparation assumptions in practice. Indeed, it is likely that the (at best) polynomial speedup of quantum state preparation (as opposed to preparing classical ℓ^2 sample access) will not be realizable with quantum hardware.

Furthermore, we discussed pseudo-inverse which inverts singular values, but in principle we could have applied any function to the singular values. Indeed, Gilyen et. al (2018) show that many quantum machine learning algorithms indeed apply polynomial functions to singular values.

In general, however, our discussion suggests that exponential quantum speedups are tightly related to problems where high-rank matrices play a crucial role (e.g. Hamiltonian simulation or Quantum Fourier Transform).

Chapter 6

Quantum Sample Complexity

6.1 PAC-learning bounds using VC Dimension for unspecified distribution

It is well known that one can tightly bound the number of samples from an unknown distribution required to PAC-learn (Section 2.2.1) an unknown concept in terms of its VC dimension (Section 2.2.6). Hence, a natural question to ask is whether "quantum samples" are more powerful in the sense of requiring less examples to achieve this goal.

Here we will explore the techniques of Arunachalam and de Wolf [5] in order to address this question. The authors show that the ideas of quantum state discrimination discussed in Section A.7.2 offer an intuitive analysis, despite resulting in a suboptimal lower bound. However, Boolean Fourier analysis (Section 2.1) applied to the Pretty Good Measurement (Section A.7.1) gives an optimal bound showing that there is no quantum advantage in terms of sample complexity when restricting to PAC learning and quantum samples as specified in the model introduced by Bshouty and Jackson [9].

In this section, we will elaborate on the former information theoretic method and demonstrate its generalizability.

6.1.1 Definitions

Quantum Learning Models: PAC Setting

A quantum example oracle $QPEX(c, D)$ acts on $|0\rangle^{\otimes n} |0\rangle$ and produces a quantum example

$$\sum_{x \in \{0,1\}^n} D(x) |x, c(x)\rangle \tag{6.1}$$

A quantum learner is given access to some copies of the state generated by $QPEX(c, D)$ and performs a POVM where each outcome is associated with a hypothesis.

A learning algorithm $\mathcal{A}$ is an (ϵ, δ)-PAC quantum learner for $\mathcal{C}$ if for every $c \in \mathcal{C}$ and distribution D, given access to the $QPEX(c, D)$ oracle, $\mathcal{A}$ outputs an h such that

$$err_D(h, c) \leq \epsilon \tag{6.2}$$

with probability at least $1 - \delta$.

The sample complexity of $\mathcal{A}$ is the maximum number invocations of the $QPEX(c, D)$ oracle, maximized over all $c \in \mathcal{C}$, distributions D, and the learners internal randomness. The (ϵ, δ)-PAC quantum sample complexity of a concept class $\mathcal{C}$ is the minimum sample complexity over all (ϵ, δ)-PAC quantum learners for $\mathcal{C}$.

Quantum Learning Models: Agnostic Setting

Notice that in the PAC setting we make an implicit assumption that the label of x is given by some $c(x)$ i.e. the labelling can be specified by a function c in our concept class $\mathcal{C}$ of consideration. It is not difficult to generalize to the "agnostic setting", as the authors do in [5] and show similar bounds. We refrain from discussing the agnostic setting given that our goal is understanding and generalized the methods used in this type of analysis.

6.1.2 Goals

We seek to show that quantum examples are not actually more powerful than classical labeled examples in the PAC model when the underlying data distribution is arbitrary. We emphasize this point on the data distribution because we plan to later detail advantages that can be reaped if the example distribution is e.g. uniform (hint: think about the Bernstein-Vazirani algorithm and a linear classifier).

In the classical case, the sample complexity of concept class $\mathcal{C}$ with VC dimension d in the PAC setting is

$$\Theta(\frac{d}{\epsilon} + \frac{\log(1/\delta)}{\epsilon})$$

where ϵ is the approximation coefficient and δ is the probability of success, as usual.

The authors indeed show that, using the quantum learning models above, the bounds are the same in the quantum case. This requires a state identification argument which uses Fourier Analysis to analyze the performance of a Pretty Good Measurement A.7.1. However, we can get close by instead using simple concepts from quantum information theory.

Of course, we know that the upper bounds in sample complexity are the same as the classical case since we can always implement a classical algorithm on a quantum computer. Hence, we seek lower bounds instead.

6.1.3 Information Theoretic Lower Bounds on Sample Complexity

VC-independent lower bounds

Lemma 6.1.1. *Let $\mathcal{C}$ be a non-trivial concept class. For every $\delta \in (0, 1/2), \epsilon \in (0, 1/4)$, a (ϵ, δ)-PAC quantum learner for $\mathcal{C}$ has sample complexity $\Omega(\frac{1}{\epsilon} \log \frac{1}{\delta})$*

Proof. Since $\mathcal{C}$ is non-trivial, we may assume there are two concepts $c_1, c_2 \in C$ defined on two inputs $\{x_1, x_2\}$ as follows:

$$c_1(x_1) = c_2(x_1) = 0$$
$$c_1(x_2) = 0, c_2(x_2) = 1$$

Consider the distribution D such that

$$D(x_1) = 1 - \epsilon$$
$$D(x_2) = \epsilon$$

For $i \in \{1, 2\}$, the state of the algorithm after T queries to $QPEX(c_i, D)$ is

$$|\psi_i\rangle = (\sqrt{1-\epsilon}\,|x_1, 0\rangle + \sqrt{\epsilon}\,|x_2, c_i(x_2)\rangle)^{\otimes T}$$

Therefore, $\langle\psi_1|\psi_2\rangle = (1-\epsilon)^T$. Since the success probability of an (ϵ, δ)-PAC quantum learner is $\geq 1 - \delta$, Corollary A.7.11.1 implies $\langle\psi_1|\psi_2\rangle \leq 2\sqrt{\delta(1-\delta)}$.
$\therefore T = \Omega(\frac{1}{\epsilon}\log\frac{1}{\delta})$ □

VC-dependent lower bounds

Theorem 6.1.2. *Let C be a concept class with $dim_{VC}(\mathcal{C}) = d+1$. Then for every $\delta \in (0, 1/2)$ and $\epsilon \in (0, 1/4)$, every (ϵ, δ)-PAC learner for $\mathcal{C}$ has sample complexity $\Omega(\frac{d}{\epsilon} + \frac{\log(1/\delta)}{\epsilon})$.*

Consider an (ϵ, δ)-PAC learner for $\mathcal{C}$ that uses T examples. The d-independent part of the lower bound, $T = \Omega(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$, was proven in Lemma 6.1.1. Hence it remains to prove $T = \Omega(d/\epsilon)$.

It suffices to show this for a specific distribution D, defined as follows. Let $S = \{s_0, s_1, \cdots, s_d\} \subseteq \{0,1\}^n$ be some $(d+1)$-element set shattered by $\mathcal{C}$. Define

$$D(s_0) = 1 - 4\epsilon \tag{6.3}$$
$$D(s_i) = 4\epsilon/d \tag{6.4}$$

for all $i \in [d]$.

Because S is shattered by $\mathcal{C}$, for each string $a \in \{0,1\}^d$, there exists a concept $c_a \in \mathcal{C}$ such that $c_a(s_0) = 0$ and $c_a(s_i) = a_i$ for all $i \in [d]$.

We essentially have two distributions of interest: the distribution which describes sampling the $\{s_i\}$ and the distribution which describes the uniform samples a. Hence, define A to be a random variable uniformly distributed over $\{0,1\}^d$. For fixed a, define $B = B_1 \otimes \cdots \otimes B_T$ as T i.i.d. quantum examples from $QPEX(c_a, D)$.

Hence, each B_i is the quantum sample

$$\sum_{i \in \{0,1\}^d} \sqrt{D(s_i)}\,|i, c_a(s_i)\rangle$$

and so the AB bipartite system can be written as

$$\frac{1}{2^d} \sum_{a \in \{0,1\}^d} |a\rangle \langle a| \otimes |\psi_a\rangle \langle \psi_a|^{\otimes T}$$

To prove this theorem, we use the following lemmas.

Lemma 6.1.3.

$$I(A:B) \geq (1-\delta)(1-H(1/4))d - H(\delta) = \Omega(d)$$

Proof. The learner will output a hypothesis $h(B) \in \{0,1\}^d$ based upon the observed samples B. Allow us to restrict our view to error on $s_1, \cdots, s_d$. In this case, the error of the hypothesis weighted by the distribution is simply the hamming distance $d_H(h(B), A)$ multiplied by $4\epsilon/d$:

$$err_D(h(B), A) = d_H(h(B), A)$$

where we are comparing with A because $c_a(A) = A$ on our restriction.

So, define indicator variable Z as indicating whether our restricted error is $\leq \epsilon$. Then, this is equivalent to requiring that

$$d_H(h(B), A) \leq d/4 \tag{6.5}$$

Since we are in search of a PAC learner, we require that $P(Z) \geq 1 - \delta$ which implies that Z has binary entropy

$$H_2(Z) \leq H_2(\delta)$$

If $h(B)$ satisfies (6.5), then A ranges over only $\sum_{i=0}^{d/4} \binom{d}{i} \leq 2^{H_2(1/4)d}$ (2.3.5) bit-strings given $h(B)$. Therefore,

$$H_2(A \mid h(B), Z = 1) \leq H_2(1/4)d$$

Furthermore, because $h(B)$ is assumed close to A we also have $H_2(A \mid B, Z = 1) \leq H_2(A \mid h(B), Z = 1)$. Now,

$$\begin{aligned} I(A:B) &= H(A) - H(A \mid B) \\ &\geq H(A) - H(A \mid B, Z) - H(Z) \\ &= H(A) - Pr(Z=1)H(A \mid B, Z=1) - Pr(Z=0)H(A \mid B, Z=0) - H(Z) \\ &\geq d - d(1-\delta)H(1/4) - d\delta - H(\delta) \\ &= \Omega(d) \end{aligned}$$

□

Lemma 6.1.4.

$$I(A:B) \leq T \cdot I(A:B_1)$$

Proof.

$$\begin{aligned}
I(A:B) &= H(B) - H(B \mid A) \\
&= H(B) - \sum_{i=1}^{T} H(B_i \mid A) && \text{(independence)} \\
&\leq \sum_{i=1}^{T} H(B_i) - \sum_{i=1}^{T} H(B_i \mid A) && \text{(A.7.5)} \\
&= \sum_{i=1}^{T} I(A:B_i) \\
&= \sum_{i=1}^{T} I(A:B_1) && (I(A:B_i) = I(A:B_1), \forall i)
\end{aligned}$$

□

Lemma 6.1.5.

$$I(A:B_1) = 4\epsilon \log(2d)$$

Proof. Since AB is a classical-quantum state, we have $I(A:B_1) = H(A) + H(B_1) - H(AB_1) = H(B_1)$, where the first equality follows from definition and the second equality uses $S(A) = d$ since A is uniformly distributed in $\{0,1\}^d$, and $S(AB_1) = d$ since the matrix $\sigma = \frac{1}{2^d} \sum_{a \in \{0,1\}^d} |a\rangle \langle a| \otimes |\psi_a\rangle \langle \psi_a|$ is block diagonal with 2^d rank-1 blocks on the diagonal. It thus suffices to bound the entropy of the singular values of the reduced state of B_1, which is

$$\rho = \frac{1}{2^d} \sum_{a \in \{0,1\}^d} |\psi_a\rangle \langle \psi_a|$$

Let $\sigma_0 \geq \sigma_1 \geq \cdots \geq \sigma_{2d} \geq 0$ be its singular values. Since ρ is a density matrix, these form a probability distribution. Note that the upper-left entry of the matrix $|\psi_a\rangle \langle \psi_a|$ is $D(s_0) = 1 - 4\epsilon$, hence so is the upper-left entry of ρ. This implies $\sigma_0 \geq 14\epsilon$.

Consider sampling a number $N \in \{0, 1, ..., 2d\}$ according to the σ-distribution. Let Z be the indicator random variable for the event $N \neq 0$, which has probability $1 - \sigma_0 \leq 4\epsilon$. Note that $H(N \mid Z = 0) = 0$, because $Z = 0$ implies $N = 0$. Also, $H(N \mid Z = 1) \leq \log(2d)$, because if $Z = 1$ then N ranges over $2d$ elements.

We now have

$$\begin{aligned}
H(\rho) = H(N) = H(N,Z) &= H(Z) + H(N|Z) \\
&= H(Z) + Pr[Z=0] \cdot H(N|Z=0) + Pr[Z=1] \cdot H(N|Z=1) \\
&\leq H(4\epsilon) + 4\epsilon \log(2d) \\
&= O(\epsilon \log(d/\epsilon)) && \text{(Using the Taylor series of the logarithm)}
\end{aligned}$$

□

Now, we are prepared to prove the theorem.

Proof. (Theorem 6.1.2) Combining these three lemmas, we have that

$$\begin{aligned} I(A:B) &\leq T \cdot I(A:B_1) \\ \Omega(\epsilon \log(d/\epsilon)) &= T \cdot 4\epsilon \\ T &= \Omega\Big(\frac{d}{\epsilon \log(d/\epsilon)}\Big) \end{aligned}$$

which comes close to optimal bound $\Omega(d/\epsilon)$ □

6.2 Exact learning bounds with uniform examples

We claimed that the information-theoretic method used in the previous section is generalizable, so we provide an example lower bound as in [6].

Consider the concept class of k-Fourier-sparse Boolean functions:

$$\mathcal{C} = \{f : \{0,1\}^n \to \{-1,1\} : \|supp(\hat{f})\| \leq k\}$$

Definition 6.2.1. *We term $(\epsilon = 0, \delta)$-PAC learning as exact learning.*

Definition 6.2.2. *Uniform examples are those produced by $QPEX(c, D)$ where D is uniform.*

Theorem 6.2.3. *$\Omega(k \log k)$ uniform quantum examples are necessary to learn the concept class of k-Fourier-sparse Boolean functions*

Proof. Sketch.

Assume for simplicity that k is a power of 2, so $\log k$ is an integer. Consider Grassmannian manifold over $\{0,1\}^n$. We can carve out the distinct subspace in $\{0,1\}^n$ with dimension $n - \log k$ and term this $\mathcal{V}$. Hence,

$$\mathcal{C} = \{c_V : \{0,1\}^n \to \{-1,1\} : c_V(x) = -1 \text{ iff } x \in V, \text{ where } V \in \mathcal{V}\}$$

Note that $|\mathcal{C}| = |\mathcal{V}|$, and each $c_V \in \mathcal{C}$ evaluates to 1 on a $(1 - 1/k)$-fraction of its domain.

We can use a similar approach to before. Let A be a random variable that is uniformly distributed over $\mathcal{C}$. Suppose $A = c_V$, then let $B = B_1 \cdots B_T$ be T copies of the quantum example

$$|\psi_V\rangle = \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x, c_V(x)\rangle$$

for c_V . The random variable B is a function of the random variable A. The following upper and lower bounds on $I(A:B)$ are similar to previous proof and we omit the details of the first two steps here.

1. $I(A:B) \geq \Omega(\log |\mathcal{V}|)$ because B allows one to recover A with high probability.
2. $I(A:B) \leq T \cdot I(A:B_1)$ using a chain rule for mutual information.

Lemma 6.2.4. $I(A:B_1) \leq O(n/k)$

Proof. Since AB is a classical-quantum state, we have

$$I(A : B_1) = S(A) + S(B_1)S(AB_1) = S(B_1),$$

where the first equality is by definition and the second equality uses $S(A) = \log|V|$ since A is uniformly distributed over $\mathcal{C}$, and $S(AB_1) = \log|V|$ since the matrix

$$\sigma = \frac{1}{\|\mathcal{V}\|} \sum_{V \in \mathcal{V}} |V\rangle \langle V| \otimes |\psi_V\rangle \langle \psi_V|$$

is block-diagonal with $|V|$ rank-1 blocks on the diagonal. It thus suffices to bound the entropy of the (vector of singular values of) the reduced state of B_1, which is

$$\rho = \frac{1}{\|\mathcal{V}\|} |\psi_V\rangle \langle \psi_V|$$

Let $\sigma_0 \geq \sigma_1 \geq \cdots \geq \sigma_{2^{n+1}-1} \geq 0$ be the singular values of ρ. Since ρ is density matrix, these form a probability distribution. Now observe that $\sigma_0 \geq 1 - 1/k$ since the inner product between $\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x, 1\rangle$ and every $|\psi_V\rangle$ is $1 - 1/k$. Let $N \in 0, 1, \cdots, 2^{n+1}1$ be a random variable with probabilities $\sigma_0, \sigma_1, \cdots, \sigma_{2^{n+1}-1}$, and Z an indicator for the event $N \neq 0$ (note that $Z = 0$ with probability $\sigma_0 \geq 1 - 1/k$). By a similar argument as in before, we have

$$\begin{aligned} S(\rho) &= H(N) = H(N, Z) = H(Z) + H(N \mid Z) \\ &= H(\sigma_0) + \sigma_0 \cdot H(N \mid Z = 0) + (1 - \sigma_0) \cdot H(N \mid Z = 1) \leq H(1/k) + \frac{n+1}{k} \leq O(\frac{n + \log k}{k}) \end{aligned}$$

using $H(N \mid Z = 0) = 0$ in the first inequality, $H(\alpha) \leq O(\alpha \log(1/\alpha))$ in the second.

Combining these three steps implies $T = \Omega(k(\log|\mathcal{V}|)/n)$. It remains to lower bound $\mathcal{V}$ (not hard).

□

□

Chapter 7

Acknowledgements

First and foremost, a special thank you to my advisor Dr. Marvian for graciously setting aside time to facilitate this journey within quantum information theory. I thank him for the pleasant and insightful conversations, endless inspiration, and impactful lessons on Physics—but also patience, communication, and perspective. I also thank the defense committee members Dr. Sun and Dr. Younes for courteously and enthusiastically participating in the thesis process.

I am grateful to the Department of Physics at Duke University and Dr. Scholberg for facilitating the opportunity to write and present this work in addition to the coursework and instruction that shaped my view of the field. I am appreciative of my academic advisor Dr. Greenside for providing my first introduction to the beauty of Quantum Physics and offering his endless support throughout my Undergraduate career.

Finally and importantly, I thank my family for their never-ending support of my endeavors.

Appendix A

Appendix

A.1 Quantum Mechanics

Definition A.1.1. *Pauli Matrices*

$$\sigma_x = X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
$$\sigma_y = Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$
$$\sigma_z = Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Definition A.1.2. *Bell States*

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}$$
$$\frac{|00\rangle - |11\rangle}{\sqrt{2}}$$
$$\frac{|10\rangle + |01\rangle}{\sqrt{2}}$$
$$\frac{|01\rangle - |10\rangle}{\sqrt{2}}$$

Definition A.1.3. *Positive Operators*

Let A be a bounded[1] linear operator on complex Hilbert space $\mathcal{H}$. The following conditions are equivalent to A being positive

1. *$A = S^\dagger S$ for some bounded operator S on $\mathcal{H}$*
2. *A is hermitian and $\langle x| A |x\rangle \geq 0$ for every $|x\rangle \in \mathcal{H}$*

[1] $\|Av\| \leq M\|v\|$ for some $M > 0$ and all $v \in \mathcal{H}$

3. the spectrum of A is non-negative

Definition A.1.4. *Trace of an Operator*
Let $\{|i\rangle\}$ be an orthonormal basis for A and so

$$\begin{aligned}\mathrm{tr}(A) &= \sum_i A_{ii} \\ &= \sum_i \langle i| A |i\rangle\end{aligned}$$

Hence, if we extend $|\psi\rangle$ to the orthonormal basis $\{|i\rangle\}$ which includes $|\psi\rangle$ as the first element (for example via the Gram-Schmidt procedure) then

$$\begin{aligned}\mathrm{tr}(A |\psi\rangle \langle\psi|) &= \sum_i \langle i| A |\psi\rangle \langle\psi|i\rangle \\ &= \langle\psi| A |\psi\rangle\end{aligned}$$

by orthonormality.

Theorem A.1.5. *Spectral Theorem*
Suppose A is a compact[2] hermitian operator (compactness ensures A has eigenvectors) on complex Hilbert space $\mathcal{H}$. Hence, there is an orthonormal basis of $\mathcal{H}$ consisting of eigenvectors of A. Each eigenvalue is in $\mathbb{R}$.

A.2 POVM measurements

POVMs are best viewed as a special case of the general measurement formalism, providing the simplest means to study post-measurement statistics without knowledge of the post measurement state. Let M_m be a projective measurement operator. Then, from Born's rule, $p(m) = \langle\psi| M_m^\dagger M_m |\psi\rangle$ so if we define $E_m \coloneqq M_m^\dagger M_m$ (which is hence positive from A.1.3) then these E_m's are sufficient for the purpose of computing probabilities. The $\{E_i\}$ satisfy the completeness relation (i.e. sum to the identity)

Definition A.2.1. *(POVM) A positive operator-valued measure (POVM) is a set $\{E_i\}$ of operators that satisfy non-negativity and completeness:*

$$\begin{aligned}\forall j : E_j &\succcurlyeq 0 \\ \sum E_j &= I\end{aligned}$$

Note that projective operators are the special case of being equivalent to their respective POVM element because $E_m = P_m^\dagger P_m = P_m$.

Proposition A.2.2. *Suppose Bob is given a quantum state chosen from a set $S = |\psi_1\rangle, \cdots, |\psi_m\rangle$ of linearly independent states. Then, we can construct a POVM $\{E_1, \cdots, E_{m+1}\}$ such that if outcome E_i occurs, $1 \leq i \leq m$, then Bob knows with certainty that he was given state $|\psi_i\rangle$.*

[2] the image under A acting on any bounded subset of $\mathcal{H}$ is a compact subset of $\mathcal{H}$

Proof. To distinguish the states we require $\langle\psi_i| E_j |\psi_i\rangle = p_i\delta_{ij}$ where $p_i > 0$ and $1 \leq i, j \leq m$.

So, we can use the Gram-Schmidt process using S as our linearly independent set. This will give us an orthonormal set $U = |\varphi_1\rangle, \cdots, |\varphi_m\rangle$ that spans the same subspace as S. Next, we can represent each $|\psi_i\rangle$ in this orthonormal basis, U. Finally, for each i we can find a vector $|\psi_i'\rangle$ in the span of U that is orthogonal to all $|\psi_j\rangle, j \neq i$. Hence, we can define $E_i = |\psi_i'\rangle\langle\psi_i'|, 1 \leq i \leq m$. Finally, take $E_{m+1} = I - \sum_m E_i$. This measurement corresponds to the "inconclusive" measurement, where Bob makes no claim of the presented state.

Creating an optimal POVM is much trickier (in the sense of minimizing the probability p_{m+1}). □

From this Proposition, we see an example use case of POVMs: a reliable way to distinguish non-orthogonal (but linearly independent) states given that we allow for the slack of an "inconclusive" measurement (E_{m+1}).

A.3 The Density Operator

An alternative formulation of quantum mechanics is possible using a tool known as the density operator.

Suppose a quantum system is one of a number of states $|\psi\rangle$ with probability p_i. We call $\{p_i, |\psi_i\rangle\}$ an ensemble of pure states. The density operator is defined

$$\rho \coloneqq \sum_i p_i |\psi_i\rangle\langle\psi_i|$$

Evolution of the density operator (under a unitary transformation) can be derived readily

$$\sum_i p_i U |\psi_i\rangle\langle\psi_i| U^\dagger = U\rho U^\dagger$$

If we perform a measurement with operator M_m with initial state $|\psi_i\rangle$ then

$$\begin{aligned} p(m \mid i) &= \langle\psi_i| M_m^\dagger M_m |\psi_i\rangle \\ &= \operatorname{tr}\left(M_m^\dagger M_m |\psi_i\rangle\langle\psi_i|\right) \end{aligned}$$

using A.1.4.

Hence, summing this conditional probability across all initial states we have

$$\begin{aligned} p(m) &= \sum_i p_i \operatorname{tr}\left(M_m^\dagger M_m |\psi_i\rangle\langle\psi_i|\right) \\ &= \operatorname{tr}\left(M_m^\dagger M_m \rho\right) \end{aligned}$$

The state after obtaining measurement result m on initial state $|\psi_i\rangle$ is

$$|\psi_i^m\rangle = \frac{M_m |\psi_i\rangle}{\sqrt{\langle\psi_i| M_m^\dagger M_m |\psi_i\rangle}}$$

and so the density operator after result m is given by

$$\begin{aligned}\rho_m &= \sum_i p(i \mid m) \left|\psi_i^m\right\rangle \left\langle\psi_i^m\right| \\ &= \sum_i p(i \mid m) \frac{M_m \left|\psi_i\right\rangle \left\langle\psi_i\right| M_m^\dagger}{\left\langle\psi_i\right| M_m^\dagger M_m \left|\psi_i\right\rangle}\end{aligned}$$

Furthermore, from Bayes' rule we have that $p(i \mid m) = \frac{p(m|i)p(i)}{p(m)}$ so we can simplify

$$\begin{aligned}\rho_m &= \sum_i \frac{p(m \mid i)p(i)}{p(m)} \frac{M_m \left|\psi_i\right\rangle \left\langle\psi_i\right| M_m^\dagger}{\left\langle\psi_i\right| M_m^\dagger M_m \left|\psi_i\right\rangle} \\ &= \sum_i \frac{p(i) \left\langle\psi_i\right| M_m^\dagger M_m \left|\psi_i\right\rangle}{\operatorname{tr}\Big(M_m^\dagger M_m \rho\Big)} \frac{M_m \left|\psi_i\right\rangle \left\langle\psi_i\right| M_m^\dagger}{\left\langle\psi_i\right| M_m^\dagger M_m \left|\psi_i\right\rangle} \\ &= \sum_i \frac{p(i) M_m \left|\psi_i\right\rangle \left\langle\psi_i\right| M_m^\dagger}{\operatorname{tr}\Big(M_m^\dagger M_m \rho\Big)} \\ &= \frac{M_m \rho M_m^\dagger}{\operatorname{tr}\Big(M_m^\dagger M_m \rho\Big)}\end{aligned}$$

A quantum state whose state $|\psi\rangle$ is known exactly is said to be in a pure state. In this case the density operator is simply $\rho = |\psi\rangle\langle\psi|$. Otherwise, ρ is in a mixed state.

A pure state satisfies $\operatorname{tr}\big(\rho^2\big) = 1$ and a mixed state $\operatorname{tr}\big(\rho^2\big) < 1$.

Imagine that our record of the result m of a measurement was lost. We would have a quantum system in the state ρ_m with probability $p(m)$ without knowing the actual value of m. Hence, the system would be described as

$$\begin{aligned}\rho &= \sum_m p(m) \rho_m \\ &= \sum_m M_m \rho M_m^\dagger\end{aligned}$$

We may wish to move away from the interpretation of the density operator as a means of describing ensembles of quantum states.

Theorem A.3.1. *An operator ρ is a density operator associated to some ensemble $\{p_i, |\psi_i\rangle\}$ if and only if it satisfies the conditions*

1. *ρ has trace equal to one*
2. *ρ is a positive operator*

The use of this theorem is that we can define a density operator to be a positive operator with trace one and hence reformulate the postulates of quantum mechanics without speaking of ensembles.

This reformulation shines when describing quantum systems whose state is not known and when describing subsystems a composite quantum system.

Remember that different ensembles of quantum states can give rise to a specific density matrix and hence one must avoid assuming that the eigenvectors and eigenvalues have special significance with regard to the represented ensemble of quantum states.

Nevertheless, there is value in discussing which ensembles give rise to the same density matrix (notably in quantum noise and error correction). Let $\left|\tilde{\psi}_i\right\rangle$ generate ρ i.e. $\rho \coloneqq \sum_i \left|\tilde{\psi}_i\right\rangle \left\langle\tilde{\psi}_i\right|$. Note that $\left|\tilde{\psi}_i\right\rangle = \sqrt{p_i}\left|\psi_i\right\rangle$ is clearly not necessarily normalized. Now, we have the following theorem.

Theorem A.3.2. *The sets $\left|\tilde{\psi}_i\right\rangle$ and $|\tilde{\varphi}_j\rangle$ generate the same ρ if and only if*

$$\left|\tilde{\psi}_i\right\rangle = \sum_j u_{ij}\left|\tilde{\varphi}_j\right\rangle$$

where the matrix with matrix elements u_{ij} is unitary.

As mentioned above, density operators are powerful tools for describing subsystems of composite systems.

Suppose we have physical systems A and B, whose state is described by ρ^{AB}. The reduced density operator for system A is defined by

$$\rho^A \coloneqq \mathrm{tr}_B(\rho^{AB})$$

where tr_B is a map of operators known as the partial trace over system B which is defined by

$$\begin{aligned}\mathrm{tr}_B(|a_1\rangle\langle a_2| \otimes |b_1\rangle\langle b_2|) &= |a_1\rangle\langle a_2|\,\mathrm{tr}(|b_1\rangle\langle b_2|)\\ &= |a_1\rangle\langle a_2|\langle b_2|b_1\rangle\end{aligned}$$

Hence, consider the Bell state $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$ and the reduced density operator of its first qubit

$$\begin{aligned}\rho &= \frac{|00\rangle+|11\rangle}{\sqrt{2}}\frac{\langle 00|+\langle 11|}{\sqrt{2}}\\ \rho^1 &= \frac{|0\rangle\langle 0|\langle 0|0\rangle + |1\rangle\langle 0|\langle 0|1\rangle + |0\rangle\langle 1|\langle 1|0\rangle + |1\rangle\langle 1|\langle 1|1\rangle}{2}\\ &= \frac{|0\rangle\langle 0| + |1\rangle\langle 1|}{2}\\ &= \frac{I}{2}\end{aligned}$$

Oddly, $\mathrm{tr}\left(\frac{I^2}{4}\right) = 1/2 < 1$ so the first qubit is in a mixed state despite the system as a whole being in a pure state. This is another hallmark of quantum entanglement.

A.4 Quantum Circuits

Reference: Chapter 4 of [32]

A.4.1 Single Qubit Operations

A single qubit in the state $a|0\rangle + b|1\rangle$ can be visualized as a point (θ, φ) on the unit sphere, where $a = \cos(\theta/2), b = e^{i\varphi} sin(\theta/2)$. This is called the Bloch sphere representation and $(\cos\varphi\sin\theta, \sin\varphi\sin\theta, \cos\theta)$ is called the Bloch vector.

Proposition A.4.1. *Let $x \in \mathbb{R}$ and A be a matrix that satisfies $A^2 = I$. We can show that*

$$\exp(iAx) = \cos(x)I + i\sin(x)A$$

X, Y, Z give rise to three useful classes of unitary matrices when they are exponentiated, the rotation operators about $\hat{x}$, $\hat{y}$, and $\hat{z}$,

$$R_x(\theta) \equiv e^{-i\theta X/2}$$
$$R_y(\theta) \equiv e^{-i\theta Y/2}$$
$$R_z(\theta) \equiv e^{-i\theta Z/2}$$

We can use the above proposition to write the above equations more conveniently.

Proposition A.4.2. *$(\hat{n}\cdot\hat{\sigma})^2 = I$, which we can use to conclude that*

$$R_n(\theta) \equiv \exp(-i\theta n\cdot\sigma/2) = \cos(\theta/2)I - i\sin(\theta/2)(n_x X + n_y Y + n_z Z)$$

Lemma A.4.3. *Suppose U is a unitary operation on a single qubit. Then there exist real numbers $\alpha, \beta, \gamma, \delta$ such that*

$$U = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)}\cos(\gamma/2) & -e^{i(\alpha-\beta/2+\delta/2)}\sin(\gamma/2) \\ e^{i(\alpha+\beta/2-\delta/2)}\sin(\gamma/2) & e^{i(\alpha+\beta/2+\delta/2)}\cos(\gamma/2) \end{bmatrix}$$

Theorem A.4.4. *Suppose U is a unitary operation on a single qubit. Then there exist real numbers $\alpha, \beta, \gamma, \delta$ such that*

$$U = e^{i\alpha} R_z(\beta) R_y(\gamma) R_z(\delta).$$

Corollary A.4.4.1. *Suppose U is a unitary gate on a single qubit. Then there exist unitary operators A, B, C on a single qubit such that $ABC = I$ and $U = e^{i\alpha} AXBXC$, where α is some overall phase factor.*

A.5 Quantum Fourier Transform

Reference: Chapter 5 of [32]

A.5.1 Quantum Fourier Transform

The quantum fourier transform on an orthonormal basis $|0\rangle, \cdots, |N-1\rangle$ is defined to be a linear operator with the following action on the basis states,

$$|j\rangle \to \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi ijk/N} |k\rangle$$

Example A.5.1. *Compute the Fourier transform of the n qubit state* $|00\cdots0\rangle$.

Proof. $|00\cdots0\rangle$ corresponds to state $|0\rangle$ in the size $N = 2^n$ computational basis. Hence, using the formula above we have

$$\begin{aligned} |0\rangle &\to \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} |k\rangle \\ &= \frac{|0\rangle + |1\rangle + \cdots + |N-1\rangle}{\sqrt{N}} \end{aligned}$$

□

We can derive an alternative product representation of the quantum fourier transform. First, represent some state $|j\rangle$ using its binary representation $j = j_1 j_2 \cdots j_n$, $j_i \in \{0, 1\}$. Then,

$$|j_1, \cdots, j_n\rangle \to \frac{(|0\rangle + e^{2\pi i 0.j_n} |1\rangle)(|0\rangle + e^{2\pi i 0.j_{n-1}j_n} |1\rangle) \cdots (|0\rangle + e^{2\pi i 0.j_1 j_2 \cdots j_n} |1\rangle)}{2^{n/2}}$$

So, define the unitary transformation

$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix} \tag{A.1}$$

Then, using the circuit below, we can see that this transformation is correctly implemented.

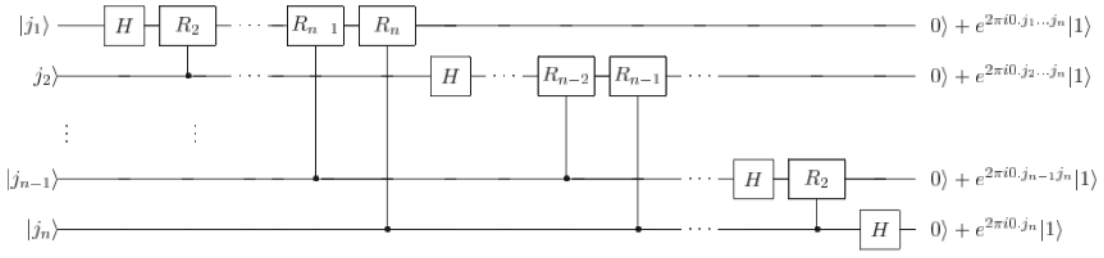


Figure 5.1. Efficient circuit for the quantum Fourier transform. This circuit is easily derived from the product representation (5.4) for the quantum Fourier transform. Not shown are swap gates at the end of the circuit which reverse the order of the qubits, or normalization factors of $1/\sqrt{2}$ in the output.

Furthermore, the gate complexity is $O(n^2)$ as opposed to $O(n2^n)$, classically.

A.5.2 Quantum Phase Estimation Algorithm

Suppose a unitary operator U has an eigenvector $|u\rangle$ with eigenvalue $e^{2\pi i\varphi}$, where the value of φ is unknown.

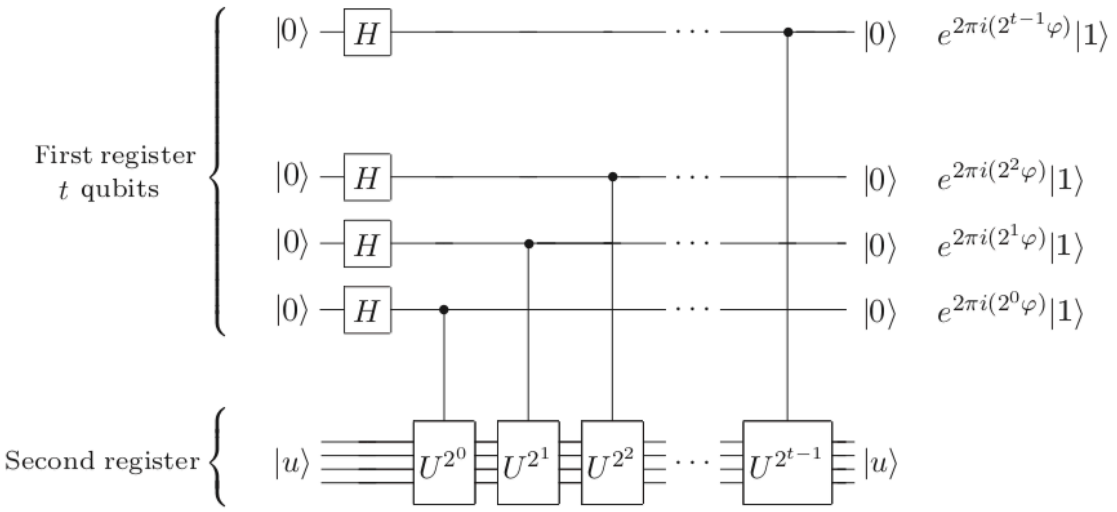


Figure 5.2. The first stage of the phase estimation procedure. Normalization factors of $1/\sqrt{2}$ have been omitted, on the right.

Then, observe that the circuit above gives the state

$$\frac{1}{2^t}(|0\rangle + e^{2\pi i 0.\varphi_t}|1\rangle)(|0\rangle + e^{2\pi i 0.\varphi_{t-1}\varphi_t}|1\rangle)\cdots(|0\rangle + e^{2\pi i 0.\varphi_1\varphi_2\cdots\varphi_t}|1\rangle)$$

Hence, we can apply the inverse QFT and get the state $|\varphi_1\cdots\varphi_t\rangle$, which is an approximation of φ, whose accuracy is dependent on the size of t.

Therefore, the complexity of this algorithm is essentially that of the inverse Fourier transform, $O(t^2)$. This assumes that each controlled-U^{2^j} operation is given by an oracle, which may not hold in practice. Furthermore, we also assume that we can prepare $|u\rangle$ efficiently, which also may not hold in practice. Hence, we often require workarounds to these problems in our applications of Phase Estimation.

Example A.5.2. *The effect of the sequence of controlled-U operations like that in the figure is to take the state $|j\rangle|u\rangle$ to $|j\rangle U^j|u\rangle$. (Note that this does not depend on $|u\rangle$ being an eigenstate of U.)*

Proof. Consider an arbitrary j in its binary representation $j_0 j_1 \cdots j_{t-1}$ where $j_i \in \{0,1\}$. Hence, for each $|j_i\rangle$, the control-U acts on $|j_i\rangle|u\rangle$ such that $|j_i\rangle|u\rangle \mapsto |j_i\rangle U^{j_i 2^i}|u\rangle$. Therefore, the final state is given by

$$\begin{aligned}|j_0\rangle \cdots |j_{t-1}\rangle U^{j_0 2^0} \cdots U^{j_{t-1}2^{t-1}} |u\rangle &= |j\rangle U^{j_0 2^0} \cdots U^{j_{t-1}2^{t-1}} |u\rangle \\ &= |j\rangle U^{j_0 2^0 + j_{t-1}2^{t-1}} |u\rangle \\ &= |j\rangle U^{j} |u\rangle\end{aligned}$$

□

A.6 Quantum Search Algorithms

For more detail, we suggest Chapter 6 of [32].

A.6.1 Grover's Algorithm and Amplitude Amplification

Suppose we wish to search a space of elements of size N. Clearly this problem is $\Omega(N)$, classically. Rather than searching the elements directly, we can focus on "searching" their indices which are labelled $[0, N-1]$ by using an oracle. So, let $x \in [0, N-1]$ and $|q\rangle$ be an ancillary qubit. Define $f(x) = 1$ if the index is the index of the solution and $f(x) = 0$ otherwise. An oracle is a unitary operation, O, which acts on the computation basis as

$$|x\rangle |q\rangle \rightarrow^O |x\rangle |q \oplus f(x)\rangle$$

Hence, if we let $|q\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} = |-\rangle$, then we can rewrite this transformation as

$$|x\rangle |-\rangle \rightarrow^O (-1)^{f(x)} |x\rangle |-\rangle$$

Grover [21] came up with a quantum algorithm that finds a solution with high probability using $O(\sqrt{N})$ oracle queries (which is known to be optimal on a quantum computer [32]).

For $N = 2^n$, the grover iteration can be given by the linear transformation $G = (2 |\psi\rangle \langle\psi| - I)O$ where $|\psi\rangle$ is the equally weighted superposition of states.

So, assuming that the number of solutions $M = 1$, Grover's algorithm essentially prepares N qubits in state $|\psi\rangle |-\rangle$ and then applies G for $\frac{\pi}{4}\sqrt{N}$ iterations.

However, if the number of solutions $M \geq 1$ is unknown, then a variant, known as amplitude amplification [8], can be used to find a solution in the solution subspace with high probability using $O(\sqrt{N/M})$ queries.

A.7 Quantum Information Theory

Standard references on this matter are [32] (we particularly use Chapter 12) and [43] (particularly Chapters 9 and 11).

Definition A.7.1. *(Quantum Entropy) Suppose that Alice prepares some quantum system A in state ρ_A (a density matrix in Hilbert space $\mathcal{H}_A$, equivalently $\rho_A \in D(\mathcal{H}_A)$). Then, the entropy $H(A)_\rho$ of the state is defined as follows*

$$H(A)_\rho := -Tr(\rho_A \log \rho_A) \tag{A.2}$$

Definition A.7.2. *(Joint Quantum Entropy) The joint quantum entropy $H(AB)_\rho$ of the density operator $\rho_{AB} \in D(\mathcal{H}_A \otimes \mathcal{H}_B)$ for a bipartite system AB follows naturally from the definition of quantum entropy*

$$H(AB)_\rho := Tr\{\rho_{AB} log \rho_{AB}\} \tag{A.3}$$

Definition A.7.3. *(Conditional Quantum Entropy) Let $\rho_{AB} \in D(\mathcal{H}_A \otimes \mathcal{H}_B)$. The conditional quantum entropy $H(A \mid B)_\rho$ of ρ_{AB} is equal to the difference of the joint quantum entropy $H(AB)_\rho$ and the marginal entropy $H(B)_\rho$:*

$$H(A \mid B)_\rho := H(AB)_\rho - H(B)_\rho \tag{A.4}$$

Definition A.7.4. *(Quantum Mutual Information) The quantum mutual information of a bipartite state $\rho_{AB} \in D(\mathcal{H}_A \otimes \mathcal{H}_B)$ is defined as follows:*

$$I(A:B)_\rho := H(A)_\rho + H(B)_\rho - H(AB)_\rho \tag{A.5}$$
$$= H(A)_\rho - H(A \mid B) \tag{A.6}$$
$$= H(B)_\rho - H(B \mid A) \tag{A.7}$$

Proposition A.7.5. *(Subadditivity of Quantum Entropy) The quantum entropy is sub-additive for a bipartite state ρ_{AB}:*

$$H(A)_\rho + H(B)_\rho \geq H(AB)_\rho$$

Proof. Corollary 11.8.1 of [43] □

Theorem A.7.6. *(Holevo's Theorem)*

Let $\{\rho_1, \rho_2, ..., \rho_n\}$ be a set of mixed states and let ρ_X be one of these states drawn according to the probability distribution $P = \{p_1, p_2, \cdots, p_n\}$.

Then, for any measurement described by POVM elements E_Y and performed on $\rho = \sum_X p_X \rho_X$, the amount of accessible information about the variable X knowing the outcome Y of the measurement is bounded from above as follows:

$$I(X:Y) \leq S(\rho) - \sum_i p_i S(\rho_i)$$

where $\rho = \sum_i p_i \rho_i$ and $S(\cdot)$ is the von Neumann entropy.

The quantity on the right hand side of this inequality is called the Holevo information or Holevo χ quantity:

$$\chi := S(\rho) - \sum_i p_i S(\rho_i)$$

A.7.1 Pretty Good Measurement (PGM)

Given a density matrix ensemble $\mathcal{E} = \{p_i, \sigma_i\}$ and a quantum state ρ we are promised that ρ is in state σ_i with probability p_i. In the general case we have $i \in [m]$ and of course $\sum_{i=1}^{m} p_i = 1$. Our goal is then to successfully identify which of the σ_i that our state ρ is actually in. This is known as Quantum Hypothesis Testing.

In some sense, thinking back to Holevo's Theorem (Theorem A.7.6), this is related to Bob attempting to access information transported from Alice that is given from the distribution above.

Hence, we perform a maximization with respect to both the probabilities on each state as well as with respect to any randomness that our approach employs. We then must choose a Quantum POVM $\{E_i\}$ that carries put a measurement and maximizes our probability of getting the state right.

So say we pick a POVM. Hence, we know that

$$\Pr(Success) = \sum_i p_i \operatorname{Tr}(\sigma_i E_i)$$

So this is the quantity that we seek to maximize.

We've shown above that the trace distance provides the solution for $m = 2$. As it turns out for $m > 2$ this is not an easy problem. However, PGM provides a sound approximation:

Intuitively, it might seem reasonable to simply choose

$$E_i = p_i \sigma_i$$

Unfortunately, then, $\sum_i E_i \neq I$. Well, one case we may think about is if we can guarantee σ_i is a the sum of pure states from an orthonormal basis. In which case, let $S = \sum_i \sigma_i$ and we choose

$$E_i = S^{1/2} \sigma_i S^{1/2}$$

Then we have

$$\operatorname{tr}(E_i \sigma_i) = \operatorname{tr}(\sigma_i) = 1$$

by orthonormality. Inspired by this, define the PGM POVM to be

$$E_i = S^{-1/2} p_i \sigma_i S^{-1/2}$$

for our original problem. Positive semidefiniteness is clear, so it remains to show that we have completeness

$$\begin{aligned} \sum_i E_i &= \sum_i S^{-1/2} p_i \sigma_i S^{-1/2} \\ &= S^{-1/2} \sum_i p_i \sigma_i S^{-1/2} \\ &= S^{-1/2} S S^{-1/2} = I \end{aligned}$$

Theorem A.7.7. *Let $\Pr_{opt}(\mathcal{E})$ by the optimal success probability for our quantum hypothesis testing problem. Define $\Pr_{PGM}(\mathcal{E})$ to be the average success probability using the PGM POVM. Then,*

$$Pr_{opt}(\mathcal{E})^2 \leq Pr_{PGM}(\mathcal{E}) \leq Pr_{opt}(\mathcal{E})$$

A.7.2 Trace Distance

Definition A.7.8. *Trace Distance.*

The trace distance $T(\cdot,\cdot)$ is a metric on the space of density operators and gives a measure of distinguishability between states. In particular, let ρ,σ be density operators,

$$\begin{aligned} T(\rho,\sigma) &= \frac{1}{2}\operatorname{Tr}\left[\sqrt{(\rho-\sigma)^2}\right] \\ &= \frac{1}{2}\sum_i |\lambda_i| \end{aligned}$$

where λ_i are the eigenvalues of Hermitian $\rho-\sigma$.

Hence, it is simply the trace norm of the positivization of the difference of matrices.

Lemma A.7.9. *For any states ρ,σ one may write $\rho-\sigma = Q-S$ where Q and S are positive operators with support on orthogonal vector spaces (Exercise 9.7 [32])*

Proof. ρ,σ are p.s.d operators. Hence, $\rho-\sigma$ is Hermitian, so we can write $\rho-\sigma = \sum_i \lambda_i |u_i\rangle\langle u_i|$ where $\{u_i\}$ is an orthonormal basis of the Hilbert space, by spectral theorem. Now, we can decompose the eigenbasis into positive and negative components. Then, $\rho-\sigma = \sum_i \lambda_i^+ |u_i\rangle\langle u_i| + \sum_j \lambda_j^- |u_j\rangle\langle u_j|$ where $\lambda_i^+ > 0, \lambda_j^- < 0$. Since, each component partitions the vector space (other than at the additive identity) by the orthogonality condition, this is a direct sum. □

Lemma A.7.10. *(Holder's Inequality for matrices)*

Given Hermitian A,B and $p,q \in [1,\infty]$ such that $\frac{1}{p}+\frac{1}{q}=1$,

$$\operatorname{tr}(AB) \le \|A\|_p \|B\|_q$$

Proposition A.7.11. *The maximum probability of distinguishing between two states with an optimal measurement is given by*

$$1/2[1+T(\rho_1,\rho_2)]$$

Proof. Say that we have the "worst-case" (equally mixed) ensemble $\{(1/2,\rho_1),(1/2,\rho_2)\}$. We seek to define a POVM $\{E_1,E_2\}$ where E_1 indicates ρ_1 and similarly for ρ_2. hence the probability of success is given by

$$\begin{aligned} \Pr_{\max} &= \max_{E_1,E_2} 1/2\operatorname{Tr}[E_1\rho_1] + 1/2\operatorname{Tr}[E_2\rho_2] \\ &= \max_{E_1,E_2} 1/2[1/2\operatorname{Tr}[(E_1+E_2)(\rho_1+\rho_2)] + 1/2\operatorname{Tr}[(E_1-E_2)(\rho_1-\rho_2)]] && \text{(linearity of trace)} \\ &= \max_{E_1,E_2} 1/2[1/2\operatorname{Tr}[(\rho_1+\rho_2)] + 1/2\operatorname{Tr}[(E_1-E_2)(\rho_1-\rho_2)]] && \text{(Completeness of POVM)} \\ &= \max_{E_1,E_2} 1/2[1+1/2\operatorname{Tr}[(E_1-E_2)(\rho_1-\rho_2)]] && \text{(trace of density operator is 1)} \end{aligned}$$

Hence, Holder's inequality says that

$$\begin{aligned}\operatorname{Tr}[(E_1 - E_2)(\rho_1 - \rho_2)] &\leq \|E_1 - E_2\|_\infty \|\rho_1 - \rho_2\|_1 \\ &\leq \|\rho_1 - \rho_2\|_1\end{aligned}$$

as desired. The first inequality follows by choosing $p = \infty, q = 1$ in Lemma A.7.10. The second inequality follows from E_1, E_2 being projector matrices and so the spectra of each satisfies $0 \leq \lambda(E_i) \leq 1$.

We can achieve this upper bound by , the optimal projection $E_1 - E_2$ is choosing E_1 to orthogonally project onto the positive eigenspace Q of $(p_1\rho_1 - p_2\rho_2)$ and E_2 the negative eigenspace S (we can do so by Lemma A.7.9). Write that the eigenvectors which span P have eigenvalues $\{\lambda_i\}_{1 \leq i \leq m}$ and Q with $\{\mu_i\}_{1 \leq i \leq m'}$. Then,

$$\begin{aligned}1/2 \operatorname{Tr}[(E_1 - E_2)(\rho_1 - \rho_2)] &= 1/2 \operatorname{Tr}[E_1(\rho_1 - \rho_2)] - \operatorname{Tr}[E_2(\rho_1 - \rho_2)] \\ &= 1/2 \sum_i \lambda_i - 1/2 \sum_i \mu_i = 1/2 \|\rho_1 - \rho_2\|_1\end{aligned}$$

as desired[3]. □

Corollary A.7.11.1. *Consider attempting to distinguish two pure states $|\psi_0\rangle, |\psi_1\rangle$. Then, we will distinguish correctly with probability at most $1/2[1 + \sqrt{1 - |\langle\psi_0|\psi_1\rangle|^2}]$.*

Equivalently, if we can distinguish between the two states w.p. $1-\delta$, then $|\langle\psi_0|\psi_1\rangle| \leq 2\sqrt{\delta(1-\delta)}$.

Proof. Applying the previous Lemma, we have maximum probability

$$1/2[1 + T(|\psi_0\rangle\langle\psi_0|, |\psi_1\rangle\langle\psi_1|)]$$

So, write $|\psi_1\rangle = \cos(\theta)|\psi_0\rangle + e^{i\varphi}\sin(\theta)|\psi_0^\perp\rangle$. Hence,

$$|\psi_1\rangle\langle\psi_1| = \cos^2(\theta)|\psi_0\rangle\langle\psi_0| + \sin^2(\theta)|\psi_0^\perp\rangle\langle\psi_0^\perp| + e^{i\varphi}\cos(\theta)\sin(\theta)|\psi_0^\perp\rangle\langle\psi_0| + e^{-i\varphi}\cos(\theta)\sin(\theta)|\psi_0\rangle\langle\psi_0^\perp|$$

Since trace is basis-independent, we can write $|\psi_0\rangle\langle\psi_0| - |\psi_1\rangle\langle\psi_1|$ in the above used orthogonal basis. This gives us characteristic polynomial

$$\begin{aligned}0 &= (1 - \cos^2(\theta) - \lambda)(-\sin^2(\theta) - \lambda) - \cos^2(\theta)\sin^2(\theta) \\ &= (\sin^2(\theta) - \lambda)(-\sin^2(\theta) - \lambda) - \cos^2(\theta)\sin^2(\theta) \\ &= -\sin^4(\theta) + \lambda^2 - \cos^2(\theta)\sin^2(\theta) \\ &= -\sin^2(\theta) + \lambda^2\end{aligned}$$

So, $\lambda = \pm|\sin(\theta)|$. Therefore, since the trace distance is the absolute sum of the eigenvalues of this difference,

[3] We did implicitly assume that $E_1 + E_2 = I$ where we could've had an additional indeterminate E_3. However, the proof would still follow in any case ([32]).

$$T(|\psi_0\rangle\langle\psi_0|, |\psi_1\rangle\langle\psi_1|) = 2|\sin(\theta)|$$

and indeed

$$\begin{aligned} |\langle\psi_0|\psi_1\rangle|^2 &= |\cos(\theta)|^2 \\ \Rightarrow \sqrt{1 - |\langle\psi_0|\psi_1\rangle|^2} &= |\sin(\theta)| \end{aligned}$$

as desired. □

Lemma A.7.12. *Let A, B, C by symmetric $d \times d$ matrices satisfying $A \succeq 0$ and $B \preceq C$. Hence,* $\mathrm{Tr}(AB) \leq \mathrm{Tr}(AC)$

Proof. Write A in its spectral decomposition $A = \sum \lambda_i |i\rangle\langle i|$, invoking Spectral Theorem (A.1.5). Hence,

$$\begin{aligned} \mathrm{Tr}(AB) &= \mathrm{Tr}\Big(\sum \lambda_i |i\rangle\langle i| B\Big) \\ &= \sum \lambda_i \mathrm{Tr}(|i\rangle\langle i| B) && \text{(linearity of trace)} \\ &= \sum \lambda_i \mathrm{Tr}(\langle i| B |i\rangle) && \text{(cyclic property of trace)} \\ &\leq \sum \lambda_i \mathrm{Tr}(\langle i| C |i\rangle) \\ &= \sum \lambda_i \mathrm{Tr}(|i\rangle\langle i| C) = \mathrm{Tr}\Big(\sum \lambda_i |i\rangle\langle i| C\Big) = \mathrm{Tr}(AC) \end{aligned}$$

□

Corollary A.7.12.1. *If $A, B \succeq 0$, then* $\mathrm{Tr}(AB) \leq \|B\|_2 \mathrm{Tr}(A)$

Proof. Note that the singular values of B coincide with the eigenvalues of B since $B^\dagger B = B^2$ and $B \succeq 0 \Rightarrow \lambda_i(B) \geq 0, \forall i$. So, let $C = \|B\|_2 I$ which then trivially satisfies $\lambda_i(C) = \lambda_{\max}(B), \forall i$ since C is the diagonal matrix with diagonal values all equal to $\lambda_{\max}(B)$. Therefore, $B \preceq C$. So, we can simply apply A.7.12 above,

$$\begin{aligned} \mathrm{Tr}(AB) &\leq \mathrm{Tr}(AC) \\ &= \mathrm{Tr}\left(A\|B\|_2 I\right) \\ &= \|B\|_2 \mathrm{Tr}(A) \end{aligned}$$

□

Bibliography

[1] Scott Aaronson. "Read the fine print". In: *Nature Physics* 11.4 (2015), p. 291.

[2] Scott Aaronson. "The learnability of quantum states". In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. Vol. 463. 2088. The Royal Society. 2007, pp. 3089–3114.

[3] Andris Ambainis. "Variable time amplitude amplification and quantum algorithms for linear algebra problems". In: *STACS'12 (29th Symposium on Theoretical Aspects of Computer Science)*. Vol. 14. LIPIcs. 2012, pp. 636–647.

[4] Srinivasan Arunachalam and Ronald de Wolf. "Guest column: a survey of quantum learning theory". In: *ACM SIGACT News* 48.2 (2017), pp. 41–67.

[5] Srinivasan Arunachalam and Ronald de Wolf. "Optimal quantum sample complexity of learning algorithms". In: *arXiv preprint arXiv:1607.00932* (2016).

[6] Srinivasan Arunachalam et al. "Two new results about quantum exact learning". In: *arXiv preprint arXiv:1810.00481* (2018).

[7] Jacob Biamonte et al. "Quantum machine learning". In: *Nature* 549.7671 (2017), p. 195.

[8] Gilles Brassard et al. "Quantum amplitude amplification and estimation". In: *Contemporary Mathematics* 305 (2002), pp. 53–74.

[9] Nader H Bshouty and Jeffrey C Jackson. "Learning DNF over the uniform distribution using a quantum example oracle". In: *SIAM Journal on Computing* 28.3 (1998), pp. 1136–1153.

[10] Yudong Cao et al. "Quantum Circuit Design for Solving Linear Systems of Equations". In: 110 (Oct. 2011).

[11] Andrew M Childs, Robin Kothari, and Rolando D Somma. "Quantum linear systems algorithm with exponentially improved dependence on precision". In: *arXiv preprint arXiv:1511.02306* (2015).

[12] Carlo Ciliberto et al. "Quantum machine learning: a classical perspective". In: *Proc. R. Soc. A* 474.2209 (2018), p. 20170551.

[13] Don Coppersmith and Shmuel Winograd. "Matrix multiplication via arithmetic progressions". In: *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. ACM. 1987, pp. 1–6.

[14] Daoyi Dong et al. "Quantum reinforcement learning". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 38.5 (2008), pp. 1207–1220.

[15] Vedran Dunjko and Hans J Briegel. "Machine learning & artificial intelligence in the quantum domain: a review of recent progress". In: *Reports on Progress in Physics* (2018). URL: http://iopscience.iop.org/10.1088/1361-6633/aab406.

[16] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm". In: *arXiv preprint arXiv:1411.4028* (2014).

[17] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. Springer series in statistics New York, 2001.

[18] Andrs Gilyn, Seth Lloyd, and Ewin Tang. "Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension". In: *arXiv preprint arXiv:1811.04909* (2018).

[19] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. "Quantum random access memory". In: *Physical review letters* 100.16 (2008), p. 160501.

[20] Gene H Golub and Charles F Van Loan. *Matrix computations*. Vol. 3. JHU Press, 2012.

[21] Lov K Grover. "A fast quantum mechanical algorithm for database search". In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. ACM. 1996, pp. 212–219.

[22] Vojtech Havlicek et al. "Supervised learning with quantum enhanced feature spaces". In: *arXiv preprint arXiv:1804.11326* (2018).

[23] Ravindran Kannan and Santosh Vempala. "Randomized algorithms in numerical linear algebra". In: *Acta Numerica* 26 (2017), pp. 95–135.

[24] Iordanis Kerenidis and Anupam Prakash. "Quantum recommendation systems". In: *arXiv preprint arXiv:1603.08675* (2016).

[25] Nathan Killoran et al. "Strawberry Fields: A software platform for photonic quantum computing". In: *Quantum* 3 (2019), p. 129.

[26] Seth Lloyd. "Quantum algorithm for solving linear systems of equations". In: *APS March Meeting Abstracts*. 2010.

[27] Seth Lloyd, Silvano Garnerone, and Paolo Zanardi. "Quantum algorithms for topological and geometric analysis of data". In: *Nature communications* 7 (2016), p. 10138.

[28] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. "Quantum algorithms for supervised and unsupervised machine learning". In: *arXiv preprint arXiv:1307.0411* (2013).

[29] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. "Quantum principal component analysis". In: *Nature Physics* 10.9 (2014), p. 631.

[30] Seth Lloyd and Christian Weedbrook. "Quantum generative adversarial learning". In: *arXiv preprint arXiv:1804.09139* (2018).

[31] Kosuke Mitarai et al. "Quantum Circuit Learning: Framework for Machine Learning with Quantum Enhanced Feature Space." In: ().

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

[33] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[34] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. "Quantum support vector machine for big data classification". In: *Physical review letters* 113.13 (2014), p. 130503.

[35] Martin Rtteler. "Quantum algorithms for highly non-linear Boolean functions". In: *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2010, pp. 448–457.

[36] Cynthia Rudin. "Support Vector Machines". In: *Duke Course Notes* ().

[37] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. "Prediction by linear regression on a quantum computer". In: *Physical Review A* 94.2 (2016), p. 022342.

[38] Changpeng Shao. "Reconsider HHL algorithm and its related quantum machine learning algorithms". In: *arXiv preprint arXiv:1803.01486* (2018).

[39] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. Vol. 1. 1. MIT press Cambridge, 1998.

[40] Ewin Tang. "Quantum-inspired classical algorithms for principal component analysis and supervised clustering". In: *arXiv preprint arXiv:1811.00414* (2018).

[41] Michael Walter. "PHYSICS 491: Symmetry and Quantum Information". In: *Stanford Lecture Notes* ().

[42] Nathan Wiebe, Daniel Braun, and Seth Lloyd. "Quantum algorithm for data fitting". In: *Physical review letters* 109.5 (2012), p. 050505.

[43] Mark M Wilde. *Quantum information theory*. Cambridge University Press, 2013.

[44] Leonard Wossnig, Zhikuan Zhao, and Anupam Prakash. "Quantum linear system algorithm for dense matrices". In: *Physical review letters* 120.5 (2018), p. 050502.

[45] VU Thi Xuan. "CR04 Report: Solving linear equations on a quantum computer". In: (2016).

[46] Zhikuan Zhao et al. "A note on state preparation for quantum machine learning". In: *arXiv preprint arXiv:1804.00281* (2018).

[47] Zhikuan Zhao et al. "Quantum algorithms for training Gaussian Processes". In: *arXiv preprint arXiv:1803.10520* (2018).

[48] Yarui Zheng et al. "Solving systems of linear equations with a superconducting quantum processor". In: *Physical review letters* 118.21 (2017), p. 210504.