Quantum Money and Scalable 21-cm Cosmology

by

Andrew Lutomirski
B.S. Physics, Stanford University, 2006
M.S. Electrical Engineering, Stanford University, 2006

Submitted to the Department of Physics in Partial Fulfillment
of the Requirements for the Degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2011

Signature of Author . . . . . .
Department of Physics
August 1, 2011

Certified by . . . . . .
Edward Farhi
Cecil and Ida Green Professor of Physics
Thesis Supervisor

Certified by . . . . . .
Max Tegmark
Professor of Physics
Thesis Co-supervisor

Accepted by . . . . . .
Krishna Rajagopal
Associate Department Head and Professor of Physics

Quantum Money and Scalable 21-cm Cosmology

by

Andrew Lutomirski

Submitted to the Department of Physics on August 1, 2011 in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Physics

Abstract

This thesis covers two unrelated topics.

The first part of my thesis is about quantum money, a cryptographic protocol in which a mint can generate a quantum state that no one can copy. In public-key quantum money, anyone can verify that a given quantum state came from the mint, and in collision-free quantum money, even the mint cannot generate two valid quantum bills with the same serial number. I present quantum state restoration, a new quantum computing technique that can be used to counterfeit several designs for quantum money. I describe a few other approaches to quantum money, one of which is published, that do not work. I then present a technique that seems to be secure based on a new mathematical object called a component mixer, and I give evidence money using this technique is hard to counterfeit. I describe a way to implement a component mixer and the corresponding quantum money using techniques from knot theory.

The second part of my thesis is about 21-cm cosmology and the Fast Fourier transform telescope. With the FFT telescope group at MIT, I worked on a design for a radio telescope that operates between 120 and 200 MHz and will scale to an extremely large number of antennas N. We use an aperture synthesis technique based on Fast Fourier transforms with computational costs proportional to $N \log N$ instead of N^2. This eliminates the cost of computers as the main limit on the size of a radio interferometer. In this type of telescope, the cost of each antenna matters regardless of how large the telescope becomes, so we focus on reducing the cost of each antenna as much as possible. I discuss the FFT aperture synthesis technique and its equivalence to standard techniques on an evenly spaced grid. I describe analog designs that can reduce the cost per antenna. I give algorithms to analyze raw data from our telescope to help debug and calibrate its components, with particular emphasis on cross-talk between channels and I/Q imbalance. Finally, I present a scalable design for a computer network that can solve the corner-turning problem.

Thesis Supervisor: Edward Farhi
Title: Cecil and Ida Green Professor of Physics
Thesis Co-supervisor: Max Tegmark
Title: Professor of Physics

Preface

For the past four years, I have worked on two unrelated projects: quantum cryptography and 21-cm radio astronomy. My two projects have essentially nothing in common, so this thesis consists of two independent parts.

Quantum computers are a new kind of computer with different and mostly unexplored powers. These computers and the communication networks that will some day connect them will change our understanding of what information can be. Manipulating quantum information will allow us to do things with information that are difficult or even impossible with ordinary classical information. I study techniques that may allow us to use quantum information to make money impossible to counterfeit or even to replace printed money entirely for some purposes.

My work in quantum cryptography is purely theoretical: I study applications of quantum technologies that do not exist today. My work is confined to the blackboard, the conference, and the computer. To keep my hands and my desire to play with real electronics busy, I have also contributed to a project at MIT to build a large radio telescope to study the early universe.

After the Big Bang, there was a long gap between when the universe cooled off enough to stop glowing and when the first stars formed and began to emit their own light. This gap is called the Epoch of Reionization. By looking far away we can see bright objects in the past, but the gap is dark; the universe was filled with cold, dark hydrogen gas, and there was almost nothing to see. If we want to directly study the EoR, we have to look at the hydrogen itself. Hydrogen emits and absorbs light at a wavelength of 21 centimeters. The signal from that emission and absorption is very faint, and no existing telescope can detect it at all, let alone see it clearly enough to learn about it. I have worked on technologies to build a larger and less expensive telescope than current techniques allow in order to detect and eventually study the 21 centimeter signal from the early universe.

In the first part of my thesis, I introduce the idea of quantum cryptography and discuss my contributions to the field of quantum money. Most of the ideas should be accessible to a general Physics or Computer Science audience, but many of the details require some knowledge of classical cryptography and the math behind quantum computing.

In the second part of my thesis, I introduce some of the techniques used to design and build large radio telescopes and the obstacles that can make those techniques expensive. I describe our approaches to working around those obstacles and designing a telescope that can eventually be made very sensitive at a small incremental cost. The physics and radio electronics communities often use different

methods and terminology, so Part II is meant to require only a basic familiarity with electronics and Fourier analysis.

Publications

Many of the ideas and some of the text in this thesis has appeared before in one form or another. The quantum cryptography chapters in particular contain little new research.

The history of quantum money (Chapter 2) draws from a review article [1] that I coauthored with Scott Aaronson, Edward Farhi, David Gosset, Avinatan Hassidim, and Jonathan Kelner. The properties of collision-free money described in Section 2.3 have been written several times [1, 2, 3, 4]; the version in my thesis is mostly from [2].

Quantum state restoration and single-copy tomography (Chapter 3) is from the extended (EPAPS / arXiv) version of [5]. Portions are copyright © 2010 The American Physical Society. The efficient attack (Section 3.6) against Wiesner's money, however, comes from [6] and the simplified version at the beginning of the chapter is mostly from [1].

The attacks against stabilizer money (Chapter 4) are from [3].

The component mixer formalism and the related theorems (Chapter 5) are all from [4], although the idea of this type of quantum money originated in [3]. Some text at the beginning of the chapter is from [2].

Quantum money from knots (Chapter 6) is from [2].

Sealed states and quantum blackmail (Chapter 7) have not been previously published, although an article should be on the arXiv soon.

The text of the radio astronomy chapters is, in contrast, mostly original. The exception is Chapter 13, which was previously published as [7].

Acknowledgments

I would like to thank a number of people and organizations who, together, made this thesis possible.

In my quantum computing work, Eddie Farhi has been almost everything I could ask for in an advisor. Peter Shor is an unmatched source of ideas, inspiration, and hints, and he deserves full credit for steering me toward quantum money in the first place. Scott Aaronson encouraged much of my work by teaching me and competing with me to develop useful protocols for quantum money. David Gosset and Avinatan Hassidim helped think about and write most of the papers that became this thesis. David Gosset also accompanied me on several interational adventures that coincided with quantum computing conferences and workshops.

My radio astronomy work would have been impossible without Max Tegmark, who suggested the whole thing. Nevada Sanchez, Chris Williams, Mike Matejek, Adrian Liu, and especially Scott Morrison worked tirelessly to prepare the proof-of-concept array for its trip to West Virginia. Nevada Sanchez and Ashley Perko worked on the second-generation array, and Eben Kunz, Jon Losh, and Katelin Schutz are still working on it. Josh Dillon, Kris Zarb Adami, and Si-Hui Tan came along on several expeditions to Maine and New Hampshire, looking at and testing sites that are far from FM radio transmitters.

My family and especially my mom, dad, and step-dad encouraged me to go to graduate school in the first place. My friends and especially John Rutherford, Elisabeth Adams, Jon Foster, and Mike Betancourt kept me sane throughout the process, even if they did everything in their power to distract me from doing real work. Saffron, Cassia and the late Rascal and Pip kept me sneezing.

Elisabeth Adams, Jon Foster, and my parents proofread surprisingly large portions of this thesis and gave me plenty of valuable suggestions.

My research and my collaborators were funded by several organizations. My MIT faculty collaborators and coauthors were supported in part by:

- The MIT Center for Theoretical Physics
- The U.S. Department of Energy under cooperative research agreement DE-FG02-94ER40818
- The W. M. Keck Foundation Center for Extreme Quantum Information Theory
- U.S. Army Research Laboratory's Army Research Office through grant number W911NF-09-1-0438

- The National Science Foundation through grant numbers CCF-0829421, CCF-0843915, CCF-0844626, AST-0607597, AST-0708534, AST-0907969, AST-0908848 and PHY-0855425
- A DARPA YFA grant
- NASA grants NAG5-11099 and NNG 05G40G
- The David and Lucile Packard Foundataion

My graduate student, post-doc, and visiting coauthors were supported in part by:

- The Natural Sciences and Engineering Research Council of Canada
- Microsoft Research
- European Project OP CE QUTE ITMS NFP 26240120009
- The Slovak Research and Development Agency under contract number APVV LPP-0430-09

Many other people, including several anonymous referees contributed valuable thoughts and feedback.

My time at MIT was funded by the Marble Presidental Fellowship in the 2007-08 academic year and by the National Defense Science and Engineering Graduate Fellowship for the 2008-11 academic years. Their generous support made it possible for me to work on two unrelated projects at the same time.

Contents

I. Quantum cryptography 15

1. What is quantum cryptography? 17
   1.1. Terminology . . . 20

2. Forty years of quantum money 25
   2.1. Classical money . . . 25
   2.2. Quantum Money . . . 26
   2.3. Collision-free quantum money . . . 28

3. State restoration: why public-key quantum money is tricky 33
   3.1. What is quantum state restoration? . . . 35
   3.2. The algorithm . . . 36
   3.3. Single-copy tomography and estimation of measurement statistics 39
      3.3.1. General tomography on a subsystem . . . 39
      3.3.2. Measurement of a subsystem in an orthogonal basis . . . 39
      3.3.3. Estimation of the statistics of any POVM . . . 40
   3.4. Single-copy tomography algorithms . . . 41
      3.4.1. Quantum state restoration . . . 41
      3.4.2. Improved algorithms . . . 42
         3.4.2.1. Alternating projections . . . 43
         3.4.2.2. Phase estimation . . . 44
      3.4.3. Performance comparison . . . 48
   3.5. A condensed-matter application of single-copy tomography . . . 49
   3.6. A faster attack against Wiesner's money . . . 49

4. Breaking Aaronson's quantum money 51
   4.1. The attack . . . 52
      4.1.1. Attacking the verifier for small ϵ . . . 52
      4.1.2. Recovering the classical secret for large ϵ . . . 52
   4.2. Details of the attack against stabilizer money for small ϵ . . . 53
      4.2.1. Generating each register's state . . . 56
      4.2.2. Review of the phase estimation algorithm . . . 59

5. Quantum money based on component mixers 61
5.1. Introduction . . . 62
5.2. Definitions . . . 64
5.3. Basic properties of component mixers . . . 66
5.4. Placing component mixer problems in the complexity zoo . . . 68
5.4.1. Inclusions . . . 68
5.4.2. Separations . . . 69
5.4.3. Conjectured separations . . . 69
5.5. A hardness result for counterfeiting quantum money . . . 70
5.6. Open problems . . . 74
5.7. Query protocols for component problems . . . 76
5.7.1. An AM query protocol for MULTIPLE BALANCED COMPONENTS 76
5.7.2. A co-AM query protocol for MULTIPLE BALANCED COMPONENTS . . . 77
5.7.3. A quantum witness for MULTIPLE COMPONENTS . . . 77
5.8. MULTIPLE COMPONENTS has exponential quantum query complexity 78

6. Quantum money from knots 79
6.1. Knots, links, and grid diagrams . . . 79
6.1.1. Knots and links . . . 80
6.1.2. The Alexander polynomial of an oriented link . . . 81
6.1.3. Grid diagrams . . . 82
6.2. Quantum money . . . 86
6.2.1. Minting quantum money . . . 86
6.2.2. Verifying quantum money . . . 88
6.2.2.1. A classical Markov chain . . . 89
6.2.2.2. The quantum verifier . . . 89
6.2.3. Security of the money scheme . . . 92
6.2.4. Why not quantum money from graphs? . . . 93
6.3. Details of the Markov chain on planar grid diagrams . . . 94

7. Sealed states and quantum blackmail 97
7.1. A bound on soundness and completeness . . . 98
7.2. Multiple compromising pictures . . . 100
7.3. CAPTCHAs . . . 101
7.4. Open questions . . . 104

II. RADIO ASTRONOMY 107

8. Introduction 109
8.1. Antenna basics . . . 111
8.2. Interferometry . . . 112

8.3. Fourier transforms . . . 113

9. Aperture synthesis and electric field gridding 115
9.1. Traditional interferometers . . . 116
9.2. An aside: polarization . . . 118
9.3. Fast Fourier transform telescope design . . . 119

10. Telescope hardware 121
10.1. Amplification and digitization . . . 122
10.1.1. Amplification . . . 122
10.1.2. Digitization and frequency conversion . . . 123
10.1.2.1. Direct sampling . . . 123
10.1.2.2. Heterodyne receivers . . . 123
10.1.2.3. Direct conversion or I/Q demodulation . . . 127
10.2. F engine . . . 128
10.3. Corner turn . . . 128
10.4. X or FFT correlator . . . 129
10.5. Proof-of-concept 16-antenna array . . . 129
10.5.1. Phase locking and sample synchronization . . . 130
10.5.2. Deploying the proof of concept . . . 131
10.6. A scalable second-generation telescope . . . 131

11. Time-domain capture and its applications 135
11.1. Capturing data over a network . . . 135
11.2. Tone estimation . . . 137

12. Real-world effects 141
12.1. Crosstalk . . . 141
12.1.1. Measuring crosstalk . . . 141
12.1.2. Canceling crosstalk . . . 144
12.1.3. Phase switching . . . 144
12.2. I/Q imbalance and quadrature phase errors . . . 146
12.2.1. Quadrature phase and filter mismatch parameters . . . 146
12.2.2. Generalized I/Q calibration . . . 148
12.2.3. Correcting I/Q errors . . . 150

13. The Butterfly Network 151
13.1. The butterfly algorithm . . . 153
13.1.1. The problem . . . 154
13.1.2. Our solution . . . 155
13.1.2.1. A mechanical solution . . . 155
13.1.2.2. The butterfly algorithm . . . 155

13.1.2.3. An even cheaper corner turner using perfect shufflers . . . 156
13.2. Implementation . . . 157
13.2.1. Layout . . . 157
13.2.1.1. Network cost . . . 157
13.2.1.2. How to further reduce the cost . . . 158
13.2.2. Technology . . . 160
13.2.2.1. Off-the-shelf hardware . . . 160
13.2.2.2. Digital ASICs . . . 160
13.2.2.3. Analog switching . . . 161
13.2.2.4. Cable technologies . . . 161

Epilogue: quantum telescopes 163

Bibliography 165

Part I.

Quantum cryptography

1. What is quantum cryptography?

For as long as people have known how to write, people have tried to keep the things they write secret. The simplest way to do this is to keep written messages away from anyone who might want to read them. People write diaries and hide them in drawers. Governments transport messages in locked briefcases and handcuff them to a courier's wrist. Spies use invisible ink, and messengers tattoo secrets on their scalp.

Anyone relying on one of these techniques is making one of two assumptions. If they have invisible ink that they believe no one can detect or a drawer that they believe no one will open, then they are assuming that they are more clever than their adversary. If they handcuff a suitcase to a courier's wrist, then they are assuming that their adversary is unwilling to do what it takes to steal the message.

These cloak-and-dagger techniques are also limited in that they require moving physical objects from one location to another. You can't send invisible ink over the radio or put a lock on an email. So people use a different approach for sending secret messages: they send the message in plain sight but encode it to make it useless to anyone who intercepts it. There are any number of ways of doing this, from the trivial to the arcane. Leonardo da Vinci wrote notes backwards so that they could only be read in a mirror. The allied forces in World War II used codetalkers, Navajo Indians who spoke a language so obscure that no one else could hope to understand it [8]. The Germans built the Enigma machine, a contraption made of gears and wheels that scrambled messages in a way that only someone with another Enigma machine could put it back together.

The idea of encoding a message in some secret way is called encryption. Its practitioners are cryptographers, and those who try to break the codes are cryptanalysts. Until the last few decades, encryption and cryptanalysis were largely in an arms race: cryptographers would develop more and more complicated and clever codes, and cryptanalysts would devise newer and cleverer tricks to break those codes. Sometimes cryptanalysts would steal an encryption machine, take it apart to learn how it ticked, and use that knowledge to crack related machines. This is, in part, how the Allies broke the Enigma cipher: they captured several Enigma machines and their codes and, from them, figured out how to crack other codes made by Enigma machines.

Since World War II, the field of cryptography has been transformed by two ideas. The first idea is that encryption schemes should not be kept secret. Modern ciphers are mathematical algorithms that take a message and use a short key to produce a

scrambled message; no one who doesn't know the key should be able to decode the message. These ciphers are published and studied worldwide, and only those that resist any attack by any known mathematical technique are used. If the Germans had believed their Enigma cipher was that strong, then they would have been willing send the Allies and their own academics an Enigma machine to play with. This concept changed cryptography from the art of making complicated devices to the science of making devices that are simple enough that other scientists could be convinced that they are secure.

The second idea is is that cryptography can do more than send secret messages between two people who share some key. This began in 1976 when Diffie and Hellman published a technique called key exchange [9]. The Diffie-Hellman key exchange protocol allows two people to generate a secret key from scratch such that even an eavesdropper who records their entire conversation cannot learn the secret key.

Diffie-Hellman key exchange is still used today, but more importantly its discovery led to an explosion of new kinds of cryptography. One new technique allows our web browsers to assure us that, when we visit our bank's website, we're really communicating with our bank and not an impostor. Other techniques make other previously unimaginable things possible. Two cryptographers can talk over the phone and flip a coin (i.e. generate a random bit, either "heads" or "tails"), each knowing that the other could not have cheated and biased the outcome. Two millionaires talking on the phone can figure out which one is richer without revealing how rich they are [10]. Voting systems can be designed that make most forms of cheating easy to detect. The possibilities are almost endless, and cryptographers discover new ones all the time. These cryptographic techniques are amazing because they work over any communication medium. Land-line calls, cell-phone calls, and internet calls can be (but sadly aren't) made untappable using the same algorithms, and computers can flip coins and compare wealth over the internet.

But cryptography that uses ordinary (or "classical") communication has limits because it must obey some basic constraints. If you want to send a message securely, then you have to assume that any eavesdropper can copy the message without being detected; your message must be secure in spite of any eavesdropper. And most obviously, anyone who has any piece of information can make as many copies of that information as they like. These constraints of cryptography are so ingrained that cryptographers rarely imagine what would be possible if they were not true.

These facts limit what cryptography can accomplish. Cryptographers would like their protocols to be secure against any attacker with any resources whatsoever, but in many cases this is impossible. Most protocols instead aim for computational security: they are secure if some mathematical assumption is true and if an attacker does not have a computer that is drastically more powerful than anyone expects. (The amount of computer power needed to break a cryptographic protocol can be made almost arbitrarily high. To break the most secure variants of the modern

flagship encryption scheme AES, you would need a computer that can perform far more operations than there are atoms in our solar system.)

But even ignoring whether a protocol can be made secure against adversaries with unlimited computing power, some things are simply impossible with classical cryptography. Take, for example, money. Abstractly, a piece of money is merely an object that people attach value to. If you have a piece of money, you can check whether it's real and you can give it to someone else. Governments make money, but counterfeiters shouldn't be able to copy it.

Nowadays, we use two kinds of cash: bills and coins. Both are physical objects that a government knows how to make, but enough of the process is secret that they are hard to copy. The tricks that make them hard to copy are almost the same cloak-and-dagger tricks that spies used to use to keep information secret: bills are printed with special inks on special paper with special marks that are only visible under a certain kind of light. There is nothing at all to prevent a determined counterfeiter from figuring out all the tricks and duplicating them.

We can ask a natural question: can we apply cryptography to cash? The answer is, for the most part, no. Any attempt to represent bills or coins as pieces of classical information will run into the constraint that information can always be copied. If a twenty dollar bill were represented by one hundred bits of information, then no matter how much mathematical cleverness protected those hundred bits, anyone who had them could copy them freely. Instead we have things like online banking and credit cards: we use cryptography to instruct other, trusted people to move money around on our behalf. But what if we wanted to do better? What if we wanted to use cryptography to design money that *anyone* can verify and transfer without help from a trusted bank? We need to find a way around the constraints of classical information.

Quantum computing promises to change the rules. On a sophisticated fiber-optic cable, or on a future quantum internet, the units of information are not sounds or letters or bits, but rather quantum states of subatomic particles. These states come in units called qubits, and qubits have strange properties. One of those properties is exactly what we want: qubits cannot be copied. The *no-cloning theorem* (see Figure 1.1) in quantum mechanics states that there is no possible device that takes any qubit and spits out two identical copies of that qubit. This is true in a deeper sense than any assumption about an adversary's computing power: the laws of quantum mechanics are simply incompatible with the existence of quantum copying machines.

Under the rules of quantum information, new kinds of cryptography are possible. The BB84 quantum key distribution protocol [11] may some day replace the Diffie-Hellman key exchange protocol and some forms of encryption. With BB84, one party (traditionally named Alice) sends a bunch of random qubits to another party, Bob. Any eavesdropper is constrained by the no-cloning theorem: if they try to copy any information along the way, they will inevitably change the message

in the process. Bob can then compare some qubits with Alice to verify that no one has eavesdropped, and the remaining qubits form a secret key that only Alice and Bob know. Alice and Bob can use that key along with well-known classical techniques to send secure messages back and forth (for example, one-time pads [12] to prevent eavesdropping and universal hashing [13] to detect attempts to tamper with the message). BB84 requires a limited form of quantum communication that is almost achievable with current technology. Everything that BB84 does can be done with classical cryptography using Diffie-Hellman key exchange and standard encryption; BB84's advantage is that it makes no assumptions about the power of the eavesdropper's computer.

In this part of my thesis, I discuss quantum cryptographic protocols that, unlike BB84, do things that are impossible without quantum communication. Quantum cryptography, unlike classical cryptography, can be applied to money; this is, in fact, the oldest idea in quantum cryptography. In the next few chapters, I discuss the history of quantum money and why designing cryptographically secure money is tantalizing but difficult. I present a new quantum computing technique that can be used to counterfeit several designs for quantum money, and I describe a few other approaches to quantum money, one of which is published, that do not work. I then present a technique that seems to be secure. The technique is based on a new mathematical object called a component mixer, and I give evidence that quantum money based on an ideal component mixer is hard to counterfeit. I describe a way to implement a component mixer and the corresponding quantum money using techniques from knot theory.

In Chapter 7, I give a second example of a quantum cryptographic protocol that is impossible with classical cryptography: a proof that a message has not been read. Imagine that you give some document to your attorney to be unsealed and published in the event of your untimely death. Later on, you decide that you do not want to the document published, and you ask your attorney to give it back. Without quantum cryptography, the best you can do is to physically seal the document, and the security is only as good as the seal. Classical cryptography cannot be used to ensure that your attorney did not break the seal and copy the document because classical cryptography cannot prevent information from being copied. For technical reasons, quantum cryptography cannot fully solve this problem either, but I show that it can be made to work in certain limited cases.

1.1. Terminology

In the coming chapters, some basic terminology will be helpful.

Cryptographic protocols are rarely 100% secure. Instead, the user of the protocol chooses the level of security as needed. In many protocols, there is a *security parameter*, often called n, which intuitively represents the amount of work that the user of the protocol is willing to do. For example, with a standard cipher, Alice and

Bob share a secret key, and the length of the key in bits is the security parameter. The larger the value of n, the harder an attacker needs to work to decode a message that Alice sends to Bob.

In a useful protocol, the amount of work that legitimate users do only increases a little bit as the security parameter increases. With ciphers, for example, Alice and Bob need to share an n bit secret key, so to increase n by one they only need to share one additional secret bit. To find the key by brute force, an attacker would have to try every possibility. With n bits, there are 2^n keys to try. If $n = 256$, then 2^n is so large that the attacker might as well not bother.

These equations are different in different protocols, but the important part is that legitimate users like Alice and Bob only need to do an amount of work that varies slowly with n while an attacker's work increases very rapidly with n. The usual criterion is that the effort that legitimate users expend is *polynomial* as a function of n and that the effort that attackers must expend is *exponential*. This is just a heuristic – if Alice and Bob had to keep track of n^{100} bits, then the protocol would not be very useful, but n^2 or n^3 is common in widely used protocols.

Sometimes, instead of discussing very large numbers, we discuss very small numbers. If Alice and Bob share an n bit key and an attacker tries once to guess the key, then the attacker has only an extremely small chance of guessing correctly. The formal way to say this is that the attacker's probability of success is *negligible* as a function of n. A function $f : \mathbb{N} \to \mathbb{R}$ is negligible if for all y there exists N_y such that $f(x) < x^{-y}$ for all $x > N_y$. Negligible functions go to zero faster than the reciprocal of any polynomial. In our example, 2^{-n} is negligible.

We will discuss several algorithms that need to select things at random. We say that $a \in_R B$ if a is a uniform random sample from B, where B is some set of possible choices.

In some cases, we will be interested in how well one random process approximates another. The measurement we use is *total variation distance*. The total variation distance between two distributions over a set D with probability density functions p and q is

$$\frac{1}{2}\sum_e |p(e) - q(e)| = \sup_{A \subseteq D} |p(A) - q(A)| .$$

The total variation distance is sometimes referred to as the statistical difference. The analogous quantum mechanical equivalent is the *trace distance*, which measures the difference between density matrices of mixed quantum states.

Quantum states – configurations of qubits – are given names that live inside the markers $|\cdot\rangle$. This symbol is called a *ket*, and the notation is due to Dirac. Greek letters are often put inside the ket as placeholders. Much as I could tell you a number and refer to the number as x, if we had quantum computers I could send you a quantum state and refer to the state as $|\psi\rangle$. Quantum states that represent money are, naturally, usually called $|\$\rangle$.

Chapter 5 delves into complexity theory. Complexity theory involves objects called *complexity classes*, and complexity classes are named by a dizzying array of acronyms and combinations of acronyms. Some, like NP, are famous; others, like $\mathrm{NP}^{\text{co-AM}}$, can be somewhat arcane. A proper introduction to complexity theory is far beyond the scope of this thesis, but it is safe to skip over the more intimidating parts of that chapter.

Imagine that someone prepares a single qubit in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and gives it to you without telling you what α and β are. Your goal is to copy that qubit. We'll call whatever algorithm you use (the supposed "quantum copy machine") C . You feed C the (unknown) qubit $|\psi\rangle$ and a blank qubit that starts in the state $|0\rangle$ and your machine needs to output the original qubit and transform the blank qubit $|0\rangle$ into a copy of $|\psi\rangle$.

You don't know in advance what α and β are, so your copy machine has to work for any values. In particular, your machine needs to work if $\alpha = 1$ and $\beta = 0$, which means

$$C\left(|0\rangle|0\rangle\right) = |0\rangle|0\rangle.$$

Similarly, your copy machine needs to work if $\alpha = 0$ and $\beta = 1$, which means

$$C\left(|1\rangle|0\rangle\right) = |1\rangle|1\rangle.$$

But quantum mechanics is linear, so any copy machine you could possibly build has to be linear as well. This means that the operator C is linear, so we can do some linear algebra:

$$\begin{aligned} C\left(|\psi\rangle|0\rangle\right) &= C\left(\left(\alpha|0\rangle + \beta|1\rangle\right)|0\rangle\right) \\ &= \alpha C\left(|0\rangle|0\rangle\right) + \beta C\left(|1\rangle|0\rangle\right) \\ &= \alpha|0\rangle|0\rangle + \beta|1\rangle|1\rangle. \end{aligned} \tag{1.1}$$

Your copy machine was supposed to copy *any* state, so the output should have been

$$\begin{aligned} &\left(\alpha|0\rangle + \beta|1\rangle\right)\left(\alpha|0\rangle + \beta|1\rangle\right) \\ &= \alpha^2|0\rangle|0\rangle + \alpha\beta|0\rangle|1\rangle + \alpha\beta|1\rangle|0\rangle + \beta^2|1\rangle|1\rangle \end{aligned}$$

If both α and β are nonzero, then the output (1.1) of your machine is not correct. This means that your copy machine C cannot possibly work.

Figure 1.1: The No Cloning Theorem

2. Forty years of quantum money

2.1. Classical money

Today, we use two basic kinds of money; both are classical.

First, there is the kind we carry around – coins, bank notes, poker chips, and precious metals. It is easy to verify their validity. You can look for the security features on paper money, you can feel coins in your hand, and, if you really know what you are doing, you can assay precious metals. All these kinds of physical money can be counterfeited, though – if you have the right equipment, you can print paper money, stamp out your own coins, or make alloys or platings that look a lot like solid precious metals. Some subway passes and copy cards are also examples of physical money – they contain small computer chips or magnetic strips that are actually worth a certain number of subway rides or copies. But these tend to be even easier to counterfeit [14]. In theory, any physical money can be counterfeited by just using the same production process that makes the original.

The way we make this kind of money secure is very much like the way we once kept secret messages secure, and it's an arms race. Fancier printed security features can be defeated by fancier counterfeiting equipment, and even the best security features are useless when people don't look for them every time they accept money as payment. In some cases, especially online or when making very large transactions, this is a big enough problem that we use another kind of money entirely.

This other kind of money is the kind that you entrust to someone else, like bank accounts and credit lines. You can carry these around with you in the form of checks, credit cards, and debit cards – portable devices that let you instruct your bank to move money on your behalf. Unlike physical money, there's no point in copying your own credit card, as it wouldn't double the amount of money in your bank. With a credit card, you can carry as much value as you want without weighing down your pockets and you can send money across the globe nearly instantaneously. This kind of money can be made cryptographically secure: we use modern, convincingly secure cryptographic techniques when we communicate with our banks over the internet or through an ATM machine. But credit cards and bank accounts have disadvantages: every time you pay someone, you need to tell your bank whom to send money to. This leaves a paper trail and doesn't work if the connection to your bank is down.

Neither of these kinds of money are ideal. For example, imagine that you are going to Las Vegas on a business trip and also want to play some high-stakes games.

You might feel conspicuous carrying a fat wad of cash. If you use a credit card, your significant other (not to mention anyone else who gets access to your bank statements) will know exactly how much money you gambled. What you really want is some kind of money that enables you to spend it without leaving a trace and to carry as much of it as you want without weighing down your pockets.

This kind of money would be digital: you could transmit it and fit as much of it as you want on some small hand-held computer. It would be self-contained, so you could pay someone without any third party being involved. And it would be cryptographically secure: attackers could never produce a counterfeit bill which passes as real money even with extraordinary resources at their disposal.

This idealized money is impossible with classical techniques because any digital piece of information which can be sent over a communication channel can also be copied. For example, if you had one hundred dollars on your computer, then you could back up your computer, spend the money, restore your computer from the backup, and spend your money again.

Can we use quantum cryptography to make this ideal kind of money? This has been an open question since the very beginning of quantum cryptography as a field.

2.2. Quantum Money

The no cloning theorem says that we should not think of qubits the same way we think about classical information. Classical information is knowledge, and once you know something you can use that knowledge without forgetting it. This is true even of bits of information that might represent pixels of a banknote. If you dissect a physical or even digital banknote and learn what's inside, you can use that knowledge to make as many new copies of that banknote as you like.

Qubits are different. If you have a qubit of information, you can manipulate it or send it to someone else. Once you do send it to someone, though, you have neither the original nor even a good description of it. This gives quantum cryptography theorists hope that that quantum information could be used as money. A mint could produce some qubits using a secret process that only it knows, and anyone else could manipulate those qubits and send them to other people as a form of payment. By the no cloning theorem, a counterfeiter would have difficulty copying those qubits.

We distinguish two broad categories of quantum money.

In the simpler version, a mint would produce a quantum bill consisting of some number of qubits. Any recipient could store the quantum bill, move it around, and send it to other people as payment for a good or service. Whenever a merchant wants to verify that the quantum bill is valid, he or she would send the qubits to the mint and the mint would check that that they were still in the correct state using some secret process. In this type of scheme, no one other than the mint

knows how to verify the money. We call this *private-key quantum money* because the key – that is, the information needed to verify the money – is private to the mint.

The other type of quantum money is *public-key* quantum money. As before, a mint would produce a quantum state and anyone could move it or spend it. The difference is that anyone would be able to verify the money themselves without communicating with the mint. Public-key money, if it could be realized, would be our ideal kind of money.

A special type of public-key quantum money is called *collision-free*. A collision-free quantum bill has a serial number, and no one, not even the mint, can make two bills with the same serial number. This is more useful than it sounds, and we discuss it in detail in Section 2.3 below.

Private-key quantum money was introduced in the first quantum cryptography paper ever written [15]. In 1969, Stephen Wiesner described a way to implement private-key quantum money in a provably secure manner. (His paper was not published until 1983.) In Wiesner's scheme, each quantum bill is a unique random quantum state that the mint labels with a serial number. The mint keeps track of the state that corresponds to the serial number of each quantum bill and it can use its knowledge of the state to verify the money. The problem with Wiesner's scheme is that it does not have any practical advantages over credit cards since the merchant must communicate with the bank to verify each transaction. So this scheme, although theoretically interesting and provably secure, would not be very useful in the real world. Wiesner's scheme is a private key quantum money scheme because the mint must keep a private secret – the complete description of the state – to use for verification.

In 1982, Bennett, Brassard, Breidbart, and Wiesner made the first attempt to design public-key quantum money [16]. Their scheme only allowed a piece of quantum money to be spent once, so they called their quantum money units "subway tokens," not bills. In hindsight, their scheme is insecure for two reasons. First, it uses a protocol called one-out-of-two oblivious transfer as a subroutine, and that subroutine turns out to be insecure [17]. Second, their subway tokens can be counterfeited by anyone who can run Shor's quantum algorithm [18] to factor large numbers. Since anyone who can use quantum money at all must have a quantum computer, anyone trying to counterfeit the money would be able to factor large numbers. In the early days of quantum cryptography, neitquaher of these attacks were obvious – Shor's algorithm was not known until more than a decade after quantum subway tokens were proposed.

The next paper about quantum money appeared in 2003 when Tokunaga, Okamoto and Imoto [19] attempted to improve Wiesner's scheme to prevent the mint from tracking each individual bill as it is used. Each bill is a random quantum state with a particular property known only to the mint. The mint publishes a process that anyone can use to convert a valid bill into a new, different random

state with the same property. By randomizing a bill before sending it to the mint for verification, a merchant prevents the mint from tracking the movement of any particular bill. This scheme has the significant disadvantage that upon discovering a single counterfeit bill, the bank is required to immediately invalidate every bill it has ever issued. This scheme is therefore not very useful.

The idea of public key quantum money gained traction in the years that followed. Aaronson proved a "complexity-theoretic no cloning theorem" which showed that even with access to a device that verifies quantum money, a counterfeiter with limited computational resources cannot copy an arbitrary state [20]. Mosca and Stebila proposed the idea of a quantum coin as distinct from a quantum bill – each quantum coin of a given denomination would be identical. Using the complexity-theoretic no cloning theorem they argued that it might be possible to implement a quantum coin protocol, but they did not give a concrete way to do it [21]. In 2009, Aaronson proposed the first concrete scheme for public key quantum money [20]. In a 2010 paper [3], my coauthors and I showed that Aaronson's scheme was insecure – the proof is in Chapter 4. At the same time, we proposed the idea of collision-free quantum money along with a general approach to implementing it. Later that year [2], we proposed a way to fill in the details, giving – as of this writing – the only published complete quantum money protocol that has not been broken. This protocol uses ideas from knot theory and is described in Chapter 6.

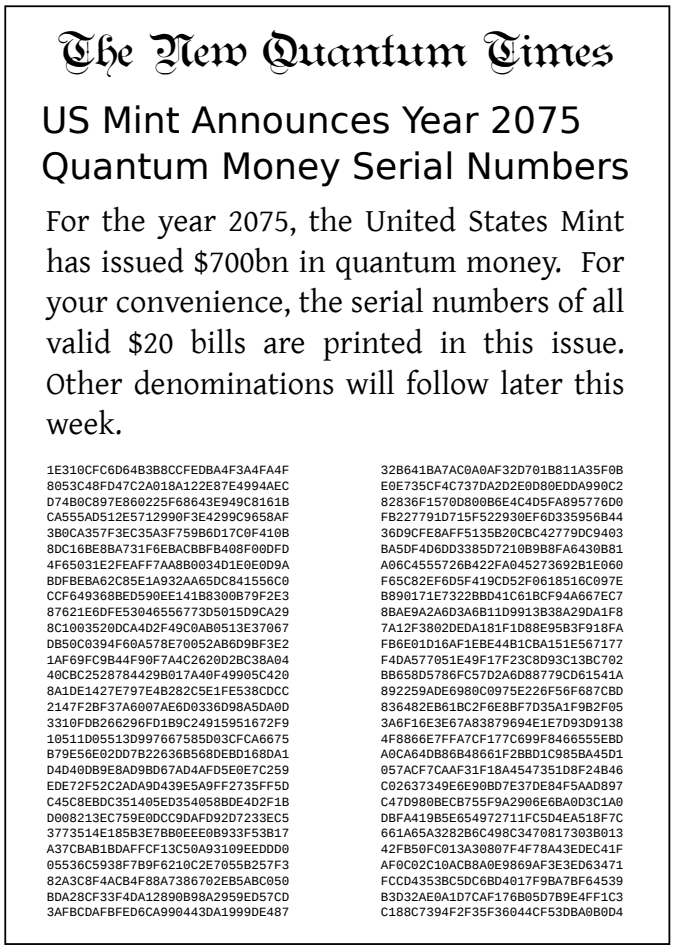
We can imagine two different ways to use quantum money for commerce. If we had the technology to print a quantum state onto a piece of paper, then we could use quantum money protocols to enhance the security of paper money against forgery. Alternatively, people could store quantum money on their personal quantum computers and use it to conduct business either in person or over a quantum internet. If small portable quantum computers were available (imagine quantum smart phones or quantum debit cards), then it would be easy to buy things with quantum money instead of paper money.

For these uses, the "quantum money" seen by an end-user would either be a file on a quantum computer or a physical piece of money with a quantum state somehow attached.

2.3. Collision-free quantum money

Collision-free quantum money is public-key quantum money with an additional requirement. Each collision-free bill has a serial number and no one, not even the mint, can produce two bills with the same serial number.

In a standard public-key quantum money scheme, a mint would know some secret process to produce a quantum bill, and everyone else would only know how to verify those bills. In contrast, a collision-free quantum money scheme does not require any secrets at all.



The New Quantum Times

US Mint Announces Year 2075 Quantum Money Serial Numbers

For the year 2075, the United States Mint has issued $700bn in quantum money. For your convenience, the serial numbers of all valid $20 bills are printed in this issue. Other denominations will follow later this week.

1E310CFC6D64B3B8CCFEDBA4F3A4FA4F
8053C48FD47C2A018A122E87E4994AEC
D74B0C897E860225F68643E949C8161B
CA555AD512E5712990F3E4299C9658AF
3B0CA357F3EC35A3F759B6D17C0F410B
8DC16BE8BA731F6EBACBBFB408F00DFD
4F65031E2FEAFF7AA8B0034D1E0E0D9A
BDFBEBA62C85E1A932AA65DC841556C0
CCF649368BED590EE141B8300B79F2E3
87621E6DFE53046556773D5015D9CA29
8C1003520DCA4D2F49C0AB0513E37067
DB50C0394F60A578E70052AB6D9BF3E2
1AF69FC9B44F90F7A4C2620D2BC38A04
40CBC2528784429B017A40F49905C420
8A1DE1427E797E4B282C5E1FE538CDCC
2147F2BF37A6007AE6D0336D98A5DA0D
3310FDB266296FD1B9C24915951672F9
10511D05513D997667585D03CFCA6675
B79E56E02DD7B22636B568DEBD168DA1
D4D40DB9E8AD9BD67AD4AFD5E0E7C259
EDE72F52C2ADA9D439E5A9FF2735FF5D
C45C8EBDC351405ED354058BDE4D2F1B
D008213EC759E0DCC9DAFD92D7233EC5
3773514E185B3E7BB0EEE0B933F53B17
A37CBAB1BDAFFCF13C50A93109EEDDD0
05536C5938F7B9F6210C2E7055B257F3
82A3C8F4ACB4F88A7386702EB5ABC050
BDA28CF33F4DA12890B98A2959ED57CD
3AFBCDAFBFED6CA990443DA1999DE487
32B641BA7AC0A0AF32D701B811A35F0B
E0E735CF4C737DA2D2E0D80EDDA990C2
82836F1570D800B6E4C4D5FA895776D0
FB227791D715F522930EF6D335956B44
36D9CFE8AFF5135B20CBC42779DC9403
BA5DF4D6DD3385D7210B9B8FA6430B81
A06C4555726B422FA045273692B1E060
F65C82EF6D5F419CD52F0618516C097E
B890171E7322BBD41C61BCF94A667EC7
8BAE9A2A6D3A6B11D9913B38A29DA1F8
7A12F3802DEDA181F1D88E95B3F918FA
FB6E01D16AF1EBE44B1CBA151E567177
F4DA577051E49F17F23C8D93C13BC702
BB658D5786FC57D2A6D88779CD61541A
892259ADE6980C0975E226F56F687CBD
836482EB61BC2F6E8BF7D35A1F9B2F05
3A6F16E3E67A83879694E1E7D93D9138
4F8866E7FFA7CF177C699F8466555EBD
A0CA64DB86B48661F2BBD1C985BA45D1
057ACF7CAAF31F18A4547351D8F24B46
C02637349E6E90BD7E37DE84F5AAD897
C47D980BECB755F9A2906E6BA0D3C1A0
DBFA419B5E654972711FC5D4EA518F7C
661A65A3282B6C498C3470817303B013
42FB50FC013A30807F4F78A43EDEC41F
AF0C02C10ACB8A0E9869AF3E3ED63471
FCCD4353BC5DC6BD4017F9BA7BF64539
B3D32AE0A1D7CAF176B05D7B9E4FF1C3
C188C7394F2F35F36044CF53DBA0B0D4

Figure 2.1: With collision-free money, the mint can publish a list of all valid serial numbers.

With collision-free quantum money there would be an algorithm that anyone can use to verify that a quantum bill matches its serial number. There would be another algorithm that *anyone* could run to produce a quantum bill. The bill that came out would have a random serial number, and the serial numbers would be long enough that the same number would be extremely unlikely to appear more than once. The mint would publish the serial number of every quantum bill it produced, and, to verify a bill, everyone would check that its serial number appears in the published list. Since no one can control the serial number of quantum bills that they produce, once the list is published no one can make a bill with a serial number in the list.

This approach has an unusual property: people do not need to trust the mint. If the mint says that it produced exactly ten billion quantum bills in a given year, anyone could check that the published list of serial numbers only has ten billion entries. This prevents the mint from printing more money than it claims.

The list of valid serial numbers would be very long, and everyone would need to make sure that their copy of the list has not been tampered with. Classical cryptography can solve both problems.

An algorithm called a *hash tree* [22] allows the mint to publish a short summary of the list of valid serial numbers; for each serial number there would be a short proof that the serial number was in lthe ist. Every user of quantum money would only need to keep track of the summary of the list.

The mint could then sign the summary with a *digital signature*. There are well-known classical digital signature protocols that allow anyone who knows the mint's "public key" to verify that a summary of a list of serial numbers was in fact published by the mint.

As an alternative to publishing a list of all serial numbers, the mint could digitally sign each serial number separately. This would make it impossible for other people to verify the number of bills produced.

In Chapter 5, I describe how a new mathematical object called a component mixer can be used to design collision-free quantum money, and I give evidence that quantum money based on component mixers can be secure. The quantum money protocol based on knot theory in Chapter 6 is an example of the component mixer design.

Formally, a collision-free quantum money scheme has two components: pieces of quantum money and an algorithm that verifies quantum money. A piece of quantum money consists of a classical serial number ℓ along with an associated quantum state $|\$_\ell\rangle$ on n qubits. The verification algorithm takes as input a quantum state $|\varphi\rangle$ and a serial number q and then decides whether or not the pair $(q, |\varphi\rangle)$ is a piece of quantum money. If the outcome is "good money" then the verifier also returns the state $|\varphi\rangle$ undamaged so it can be used again. The formal requirements are:

1. There is a polynomial-time algorithm that produces both a quantum money state $|\$_\ell\rangle$ and an associated serial number ℓ. The quantum money-producing algorithm must have a negligibly small probability of failure.

2. Running the verification algorithm with inputs ℓ and $|\$_\ell\rangle$ returns "good money" and does not damage $|\$_\ell\rangle$. Furthermore, anyone with access to a quantum computer (for example a merchant) can run the verification algorithm. The verification algorithm must have a negligibly small probability of failure.

3. Given one piece of quantum money $(\ell, |\$_\ell\rangle)$, it is hard to generate a quantum state $|\psi\rangle$ on $2n$ qubits such that each part of $|\psi\rangle$ (along with the original serial number ℓ) passes the verification algorithm.

4. *(Collision-freedom)* It is hard to generate a quantum money state with any particular label more than once. That is, it it hard to generate any quantum state of the form $|\$_\ell\rangle \otimes |\$_\ell\rangle$. This property implies property 3.

When we say that something is "hard" we mean that no polynomial-time algorithm can do it with non-negligible probability of success.

When the mint produces a series of quantum bills, it produces a set of serial numbers and matching quantum bills. In our quantum money scheme, the mint does not choose the serial numbers in advance; rather, they are produced by a random process. A rogue mint running the same algorithm as the mint can produce a new set of money pairs, but with high probability none of the serial numbers will match those that the mint originally produced.

3. State restoration: why public-key quantum money is tricky

Wiesner's original quantum money scheme [15] is straightforward. To produce a quantum bill using n qubits, the mint first chooses n one-qubit states randomly drawn from the set $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. The mint then assigns that state a (classical) serial number. A piece of quantum money consists of the n qubit state and its serial number. The mint keeps a list of all serial numbers issued as well as a description of which state corresponds to which serial number. When a merchant receives a quantum bill, the merchant sends that bill to the mint for verification. The mint looks up the serial number and retrieves the description of the corresponding quantum state. Then the mint verifies that the given state is the state that goes with the attached serial number.

This kind of money cannot be forged by someone outside the mint. Since a would-be forger has no knowledge of the basis that each qubit was prepared in, the quantum no cloning theorem says that he or she cannot reliably copy the n qubit quantum state.

Two things go wrong if you try to make public-key quantum money based on Wiesner's scheme.

The first is just a technicality. The no cloning theorem says that there is no way at all to copy a completely unknown quantum state, so if you are given one of Wiesner's quantum bills then there is nothing you can do to copy it. If, on the other hand, you are given Wiesner's quantum bill and access to a machine that will tell you whether a quantum bill is valid, you can copy the bill. Simply prepare a new random quantum state and feed it to the verification machine. With probability 2^{-n} you will get lucky and the machine will accept the bill. If it does, you just copied the bill. If the security parameter n is large enough, the Earth will probably no longer exist by the time you successfully copy of bill. The complexity-theoretic no cloning theorem [20] tells us that, if the quantum money state is completely random, then there is no clever trick that can speed this process up. Copying a bill using only a verification machine will take an exponential number of tries.

The second problem is real. The complexity theoretic no cloning theorem applies only to completely random states, and Wiesner's quantum money is a product state. Product states can be copied very quickly using only a verification machine.

A counterfeiter would start out with a single valid quantum bill. This bill is the quantum state

$$|\psi\rangle = |\psi_1\rangle|\psi_2\rangle...|\psi_n\rangle$$

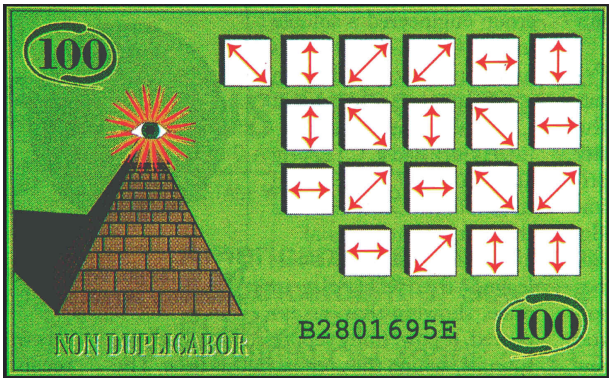


Figure 3.1: Wiesner's quantum money is a bunch of qubits along with a serial number. From [23]. Reprinted with permission from AAAS.

along with the associated serial number. The counterfeiter does not know what $|\psi_1\rangle$, $|\psi_2\rangle$, and so on are. The counterfeiter produces a random one-qubit state $|\varphi_1\rangle$, and, setting aside the first qubit $|\psi_1\rangle$ of the original bill, feeds the state

$$|\psi'\rangle = |\varphi_1\rangle|\psi_2\rangle...|\psi_n\rangle$$

into the verification machine. If the bill $|\psi'\rangle$ turns out to be valid (this happens with probability ½), then the state $|\psi'\rangle$ collapses to $|\psi\rangle$. Now the counterfeiter possesses both $|\psi\rangle$ and the original qubit $|\psi_1\rangle$ that was set aside, and so he or she has succeeded in copying the first qubit $|\psi_1\rangle$. On the other hand, if the bill $|\psi'\rangle$ turns out not to be valid, then the state of the bill collapses to

$$|\psi_1^\perp\rangle|\psi_2\rangle...|\psi_n\rangle$$

where $|\psi_1^\perp\rangle$ is the one-qubit state orthogonal to $|\psi_1\rangle$. The states of qubits 2 through n have not been changed by this process. So the counterfeiter can then throw away $|\psi_1^\perp\rangle$, replace it with a random state, and try again. After an average of two tries, the counterfeiter will have copied the first qubit of the quantum bill. Then the counterfeiter can repeat this whole procedure to copy the second qubit, the third qubit, and so on until all n qubits have been copied.

This attack works against any kind of public-key quantum money in which the quantum bill is a product state. In the specific case of Wiesner's quantum money, an even faster attack is possible – see Section 3.6.

These attacks are only possible because Wiesner's money is a product state. If the money were an entangled state, then removing one qubit of the state and verifying the remaining bits could collapse the removed qubit. It turns out that the attack generalizes to a process called quantum state restoration, and any secure public-key quantum money scheme must resist attacks based on it.

3.1. What is quantum state restoration?

Quantum mechanics places constraints on what can be done only with a single copy of an unknown state. The no-cloning theorem says that it is impossible to copy such a state. Measuring an observable on an unknown state generically damages it. Learning the full description of a state or even the description of a small piece of it cannot be done with only a single copy of it.

We are interested in the additional power given by the ability to verify a state. Given a single copy of an unknown quantum state $|\psi\rangle$ and a verifier, that is a black box (or quantum circuit) which measures the operator $P = |\psi\rangle\langle\psi|$, the no-cloning theorem no longer applies. In this setting, we present novel algorithms that can copy small parts of the state and make measurements on $|\psi\rangle$ without damaging the state. One situation where such a verifier exists is when $|\psi\rangle$ is the unique ground state of a particular gapped local Hamiltonian which we know. Measuring the energy of $|\psi\rangle$ gives the ground state energy E_0. We can then use E_0 and the Hamiltonian H to verify whether any state has energy E_0.[1]

To understand quantum state restoration, first consider a classical problem. Suppose that there is some unknown n-bit string $z = z_A z_B$, where z_A is the first $n - k$ bits of z and z_B is the last k bits. Suppose further that there is a function

$$f(x) = \begin{cases} 1 & \text{if } x = z \\ 0 & \text{otherwise} \end{cases}$$

on n-bit strings that tests whether they are equal to z. If we are given z_A and the ability to evaluate f, we can find z by randomly guessing: we pick a random k-bit string x_B and evaluate $f(z_A x_B)$, repeating until we get $f = 1$. This finds z in expected time 2^k.

Quantum state restoration is a straightforward quantum generalization of this classical algorithm, and, surprisingly, it works even on entangled states. If $|\psi\rangle$ lives in the Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$, our algorithm takes as input the part of $|\psi\rangle$ that lives in subsystem A and uses P to produce as output the state $|\psi\rangle$ in expected time $O(\text{poly}(\dim \mathcal{H}_B))$. It works by randomly guessing the part of $|\psi\rangle$ that lives in subsystem B and measuring P. On a successful iteration (i.e. if the measurement outcome is 1), then $|\psi\rangle$ is recovered. On a failed iteration, there is minimal damage to the part of the state in subsystem A and we can try again.

This can be used to copy small subsystems of $|\psi\rangle$: if $|\psi\rangle$ has the reduced density matrix ρ_B on a small subsystem B, we can set aside subsystem B and then use state restoration to extend subsystem A to the full state $|\psi\rangle$. We are left with $|\psi\rangle$ and a mixed state ρ_B. If we use this to obtain multiple copies of ρ_B, we can perform tomography on subsystem B. We call this application single-copy tomography, and

[1] Verification is not the same as measuring the energy. One way to verify the state is to apply phase estimation, compute an indicator of whether the energy has the right value, uncompute the phase estimation step, and measure the indicator.

we give two more specialized algorithms to do the same thing. All these algorithms have running time polynomial in the dimension of subsystem B. We also give a reduction from estimating the statistics of a general POVM measurement (even if it includes non-commuting operators) to single-copy tomography, with running time polynomial in the number of POVM operators.

3.2. The algorithm

Quantum state restoration takes as input a large subsystem of a state $|\psi\rangle$ (this subsystem could be, for example, the first $n-k$ qubits of the n qubit state $|\psi\rangle$) and, using the ability to measure the projector $P = |\psi\rangle\langle\psi|$, reconstructs the full state $|\psi\rangle$.

Theorem 1. *Suppose that $|\psi\rangle$ is an unknown quantum state in a Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$ and we are given oracle access to a coherent measurement of $P = |\psi\rangle\langle\psi|$ (that is, the oracle performs the operation $P \otimes \mathbb{I} + (1-P) \otimes \sigma_x$ on the original Hilbert space plus a single-qubit ancilla). Then there exists an efficient quantum algorithm that takes as input a mixed state in $\mathcal{H}_A$ with density matrix $Tr_B |\psi\rangle\langle\psi|$ and outputs $|\psi\rangle$. This algorithm makes an expected number $O\left((\dim \mathcal{H}_B)^2\right)$ of calls to the measurement oracle.*

The idea is that any state $|\psi\rangle$ on a Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$ (where d is the dimension of $\mathcal{H}_B$) can be Schmidt decomposed as

$$|\psi\rangle = \sum_{i=1}^{\chi} \sqrt{p_i} |u_i\rangle |v_i\rangle$$

where χ is the Schmidt rank of $|\psi\rangle$ (note that $\chi \leq d$). If we start with the state $|\psi\rangle$ and set aside the part that lives on $\mathcal{H}_B$, then we are left with the mixed state $\rho_A = \mathrm{Tr}_B |\psi\rangle\langle\psi|$, which has all of its support on the Schmidt basis span $\{|u_i\rangle\}$. From ρ_A, we can construct the state $\rho_A \otimes \frac{I}{d}$ on $\mathcal{H}_A \otimes \mathcal{H}_B$. We now measure the projector P. If we obtain the outcome 1, then we are left with the state $|\psi\rangle$, If not, we discard (i.e. trace out) $\mathcal{H}_B$, leaving a state on $\mathcal{H}_A$ that *still* has all of its support on the Schmidt basis. We then try again until we obtain the outcome 1. If all the p_i are equal, then each attempt succeeds with probability $\frac{1}{\chi d}$, and the entire algorithm finishes in an expected number of iterations χd. For general values $\{p_i\}$, the expected running time is still exactly χd, although the distribution of the running time becomes more complicated.

We now summarize the quantum state restoration algorithm.

1. Start with the state $|\psi\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$ and set aside the part of $|\psi\rangle$ that lives in subsystem B. We are left with the mixed state

$$\rho_A = \mathrm{Tr}_B |\psi\rangle\langle\psi|.$$

2. Add a random state on subsystem B. The state is now
$$\rho_A \otimes \frac{\mathbb{I}}{d}.$$
3. Measure the projector $P = |\psi\rangle\langle\psi|$. If the outcome is $+1$ then you are done: you still have the original copy of subsystem B that you set aside and you have recovered the state $|\psi\rangle$. If not, discard subsystem B and repeat from step 2.

We now show that the expected running time of this algorithm is $\chi \cdot d \leq d^2$ (measured in number of uses of P).

In the simple case where all of the p_i are equal, then the initial state ρ_A is the fully mixed state over the span $\{|u_i\rangle\}$. In this case, if you measure 0 in step 3, the density matrix left in register A after discarding register B is unchanged. The algorithm terminates with probability $\frac{1}{\chi \cdot d}$ on each iteration, finishing in an expected number of iterations $\chi \cdot d$. If the p_i are not all equal, then the algorithm can reach bad states where most of the weight is on low-weight elements of the Schmidt basis. When this happens, the chance of success on any given iteration drops (see Fig. 3.2 for an extreme example), but the probability of reaching these bad states decreases with the corresponding p_i. Surprisingly, these effects exactly cancel, and the expected number of iterations required to restore the state is $\chi \cdot d$ regardless of the values of the p_i.

To prove this, we define two maps

$$F_0(\sigma) = \mathrm{Tr}_B \left[(1 - |\psi\rangle\langle\psi|) \left(\sigma \otimes \frac{\mathbb{I}}{d} \right) (1 - |\psi\rangle\langle\psi|) \right]$$
$$F_1(\sigma) = \mathrm{Tr}_B \left[|\psi\rangle\langle\psi| \left(\sigma \otimes \frac{\mathbb{I}}{d} \right) |\psi\rangle\langle\psi| \right].$$

Here $F_b(\sigma)$ is the unnormalized density matrix obtained by measuring P on the state given by the density matrix σ, conditioned on the measurement outcome $b \in \{0, 1\}$. The probability of obtaining a sequence of measurement outcomes $b_1, b_2, \ldots, b_m$, starting with the state σ is then given by

$$\Pr\left[\{b_1, b_2, b_3, ..., b_m\} \,\middle|\, \sigma\right] = \mathrm{Tr}[F_{b_m} \circ \cdots \circ F_{b_1}(\sigma)], \tag{3.1}$$

which can be seen by induction:

$$\begin{aligned}
\Pr\left[\{b_1, b_2, b_3, ..., b_m\} \,\middle|\, \sigma\right] &= \Pr\left[b_m \,\middle|\, \sigma, \{b_1, b_2, b_3, ..., b_{m-1}\}\right] \\
&\quad \times \Pr\left[\{b_1, b_2, b_3, ..., b_{m-1}\} \,\middle|\, \sigma\right] \\
&= \mathrm{Tr}\, F_{b_m} \left(\frac{F_{b_{m-1}} \circ \cdots \circ F_{b_1}(\sigma)}{\mathrm{Tr}\left(F_{b_{m-1}} \circ \cdots \circ F_{b_1}(\sigma)\right)} \right) \\
&\quad \times \mathrm{Tr}\left(F_{b_{m-1}} \circ \cdots \circ F_{b_1}(\sigma)\right) \\
&= \mathrm{Tr}\, F_{b_m}\left(F_{b_{m-1}} \circ \cdots \circ F_{b_1}(\sigma)\right).
\end{aligned}$$

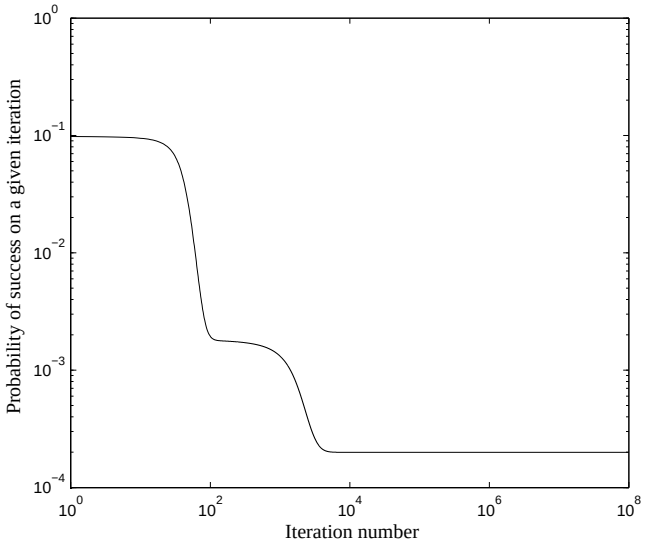


Figure 3.2: Each quantum state restoration iteration can be less likely to succeed than the previous iteration. The Y axis is the probability of restoring the state on a given iteration conditioned on all previous iterations failing. Conditioned on failing every time, the first two flat regions are metastable states and the third is stable. In this graph, $|\psi\rangle = \sqrt{1-10^{-2}-10^{-4}}|0\rangle_A|0\rangle_B + \sqrt{10^{-2}}|1\rangle_A|1\rangle_B + \sqrt{10^{-4}}|2\rangle_A|2\rangle_B$, $\dim \mathcal{H}_B = 10$, and the expected number of iterations required is 30.

We can use this equation to write an explicit formula for the expected number of measurements $T(\sigma)$, starting with the state σ:

$$\begin{aligned} T(\sigma) &= \sum_{k=1}^{\infty} k \cdot \Pr\left[\{\underbrace{0,0,\ldots,0}_{k-1},1\} \,\middle|\, \sigma\right] \\ &= \sum_{k=1}^{\infty} k \cdot \mathrm{Tr}\left[F_1 \circ F_0 \circ \cdots \circ F_0(\sigma)\right]. \end{aligned} \tag{3.2}$$

As written, this formula is difficult to evaluate, but we can see that it is linear in σ. We are interested in the quantity $T(\rho_A)$, which we expand as

$$T(\rho_A) = \sum_{i=1}^{\chi} p_i T(|u_i\rangle\langle u_i|). \tag{3.3}$$

We expand $T(|u_i\rangle\langle u_i|)$ by conditioning on the outcome of the first measurement:

$$\begin{aligned} T\left(|u_i\rangle\langle u_i|\right) &= \Pr\left[1 \,\middle|\, |u_i\rangle\langle u_i|\right] + \Pr\left[0 \,\middle|\, |u_i\rangle\langle u_i|\right] \left(1 + T\left(\frac{F_0(|u_i\rangle\langle u_i|)}{\Pr\left[0 \,\middle|\, |u_i\rangle\langle u_i|\right]}\right)\right) \\ &= 1 + T\left(F_0(|u_i\rangle\langle u_i|)\right) \\ &= 1 + T\left(|u_i\rangle\langle u_i| - 2\frac{p_i}{d}|u_i\rangle\langle u_i| + \frac{p_i}{d}\sum_{j=1}^{\chi} p_j |u_j\rangle\langle u_j|\right) \\ &= 1 + \left(1 - 2\frac{p_i}{d}\right) T\left(|u_i\rangle\langle u_i|\right) + \frac{p_i}{d}\sum_{j=1}^{\chi} p_j T\left(|u_j\rangle\langle u_j|\right). \end{aligned}$$

Using (3.3), this can be transformed into

$$2p_i T\left(|u_i\rangle\langle u_i|\right) - p_i T(\rho_A) = d.$$

Summing both sides over $i = 1, \ldots, \chi$ using $\sum p_i = 1$ and (3.3) again, we obtain

$$T(\rho_A) = \chi \cdot d,$$

which is the desired result. This proves theorem 1.

3.3. Single-copy tomography and estimation of measurement statistics

We expect that quantum state restoration will most commonly be used to perform tomography on a single copy of a verifiable quantum state. We can perform several different types of tomography, and we give algorithms for some types that are faster than quantum state restoration.

3.3.1. General tomography on a subsystem

In the simplest case, we have a single copy of an unknown state $|\psi\rangle$ and access to the measurement $P = |\psi\rangle\langle\psi|$ and we would like to estimate properties of the density matrix $\rho_B = \mathrm{Tr}_A\, |\psi\rangle\langle\psi|$ for a subsystem B. We can do this by using quantum state restoration to prepare many unentangled states, each with (independent) density matrices ρ_B. We can then use any standard state tomography algorithm on these states.

3.3.2. Measurement of a subsystem in an orthogonal basis

For many applications, it is sufficient to estimate the probabilities

$$q_i = \mathrm{Tr}\left[|i\rangle_B\langle i|_B |\psi\rangle\langle\psi|\right]$$

of obtaining the outcome i if one were to measure subsystem B of $|\psi\rangle$ in the orthonormal basis $\{|i\rangle_B\}$. Quantum state restoration can sample these probabilities directly. We discuss this application in section 3.4.1.

In sections 3.4.1 and 3.4.2, we present two other specialized algorithms to compute these probabilities. Both algorithms measure the q_i one at a time by considering the statistics of the two-outcome measurements $\{|i\rangle_B\langle i|_B, \mathbb{I} - |i\rangle_B\langle i|_B\}$, and both are based on previously presented schemes for amplifying QMA verifiers [24, 25].

In each case, we fix a precision $\delta > 0$ and an error probability $\varepsilon > 0$ and compute the running time to produce estimates q_i^{est} such that

$$|q_i^{\text{est}} - q_i| < \delta$$

for all i with probability at least $1 - \varepsilon$.

3.3.3. Estimation of the statistics of any POVM

We can use any of our algorithms to estimate the statistics of a general measurement (on the complete state, not just a subsystem). This is because a general POVM measurement can be reduced to a measurement of a subsystem in an orthogonal basis, as we now review. Given an efficiently implementable POVM $\{E_i\}$ where $i \in \{1, \ldots, d\}$, we can implement a unitary operator U such that

$$U\left(|\varphi\rangle_A|1\rangle_B\right) = \sum_{i=1}^{d} \left(\sqrt{E_i}|\varphi\rangle_A\right)|i\rangle_B$$

for any state $|\varphi\rangle$. If we work in a two-register Hilbert space, where register A can hold $|\varphi\rangle$ and register B has dimension d, then the probability of measurement outcome i when the POVM is measured on $|\varphi\rangle$ is equal to

$$\langle\varphi|E_i|\varphi\rangle = \text{Tr}\left[\rho_B|i\rangle_B\langle i|_B\right]$$

where $\rho_B = \text{Tr}_A\left[U|\varphi\rangle_A|1\rangle_B\langle 1|_B\langle\varphi|_A U^\dagger\right]$. If we define

$$\begin{aligned} |\psi\rangle &= U|\varphi\rangle_A|1\rangle_B \\ P' &= |\psi\rangle\langle\psi| = UPU^\dagger \end{aligned}$$

then $|\psi\rangle$ can be efficiently prepared (given $|\varphi\rangle$) and P' can be efficiently measured. Now we can use any of the algorithms to estimate the measurement statistics of subsystem B of $|\psi\rangle$ using the projector P' in the computational basis (that is, any of the algorithms below) to estimate the probabilities $\langle\varphi|E_i|\varphi\rangle = \text{Tr}\left[|i\rangle\langle i|\rho'_B\right]$. After estimating the probabilities, we uncompute U to recover the initial state $|\varphi\rangle$. We summarize this ability with the following theorem.

THEOREM 2. *Suppose that $|\varphi\rangle$ is an unknown quantum state and we are given oracle access to a coherent measurement of $P = |\varphi\rangle\langle\varphi|$ (that is, the oracle performs the operation $P \otimes \mathbb{I} + (1-P) \otimes \sigma_x$ on the original Hilbert space plus a single-qubit ancilla). Fix $0 < \varepsilon < 1$, $\delta > 0$, and an efficiently implementable d-outcome POVM given by operators $\{E_i\}$. Then there exists an efficient quantum algorithm that takes as input a single copy of $|\varphi\rangle$ and outputs an undamaged copy of $|\varphi\rangle$ along with estimates q_i^{est} such that*

$$|q_i^{est} - \langle\varphi|E_i|\varphi\rangle| < \delta$$

for all i with probability at least $1-\varepsilon$. This algorithm uses an expected number $O\left(\frac{d}{\delta}\log\left(\frac{d}{\varepsilon}\right)\right)$ calls to the measurement oracle and the POVM.

The algorithm which achieves this running time is given in section 3.4.2.2.

If we want to perform tomography on a subsystem of $|\varphi\rangle$, we can use theorem 2 to estimate an informationally complete POVM on that subsystem.

3.4. Single-copy tomography algorithms

We give three single-copy tomography algorithms. They all implement measurements in an orthogonal basis as in Section 3.3.2; the technique in Section 3.3.3 will extend any of them to general POVMs.

3.4.1. Quantum state restoration

In this section we consider the running time of estimating the probabilities $q_i = \mathrm{Tr}\left[\rho_B |i\rangle_B\langle i|_B\right]$ on a given state $|\psi\rangle$ using quantum state restoration. We do this by repeatedly measuring register B and then restoring the state. Let m_i be the number of times we observe outcome i in N trials. Our estimate of q_i is

$$q_i^{\text{est}} = \frac{m_i}{N}.$$

For the j^{th} observation, let $x_{i,j} \in \{0,1\}$ indicate whether the outcome of that observation was i. For fixed i, the $x_{i,j}$ are independent. To obtain a bound on the error $|q_i^{\text{est}} - q_i|$, we use Hoeffding's inequality [26], which for a sequence of N independent and identically distributed random bits $x_{i,j}$ with mean value $\mathbb{E}_j[x_{i,j}] = q_i$ implies that

$$\Pr\left[\left|\frac{1}{N}\sum_{j=1}^{N} x_{i,j} - q_i\right| \geq \delta\right] \leq 2e^{-2N\delta^2}, \text{ for any } \delta > 0. \tag{3.4}$$

So

$$\Pr\left[|q_i^{\text{est}} - q_i| \geq \delta\right] \leq 2e^{-2N\delta^2}$$

for each i individually, and, by a union bound,

$$\Pr\left[|q_i^{\text{est}} - q_i| \geq \delta \text{ for any } i\right] \leq 2de^{-2N\delta^2}.$$

Choosing $N = \left\lceil \frac{1}{2\delta^2} \ln \frac{2d}{\varepsilon} \right\rceil$ makes the right hand side $\leq \varepsilon$. Each of the N repetitions of quantum state restoration takes an expected time $\chi \cdot d$, so the total expected number $\mathbb{E}[M_{\text{SR}}]$ (where the subscript stands for "state restoration") of uses of P is

$$\mathbb{E}[M_{\text{SR}}] = \chi \cdot d \left\lceil \frac{1}{2\delta^2} \ln \frac{2d}{\varepsilon} \right\rceil.$$

3.4.2. Improved algorithms

In this section we describe two other algorithms which can be used for single-copy tomography. Both of these approaches are based on Jordan's lemma [27]. The algorithms we discuss in this section are based on the QMA amplification schemes of Marriott and Watrous [24] and Nagaj et al. [25].

To use these algorithms, we fix $i \in \{1, \ldots, d\}$ and we will estimate

$$q_i = \text{Tr}\left[\rho_B |i\rangle_B \langle i|_B\right].$$

We repeat this for each value of i.

We begin by defining the projector

$$Q_i = |i\rangle_B \langle i|_B$$

and the states

$$\begin{aligned} |v_i\rangle &= \frac{1}{\sqrt{q_i}} Q_i |\psi\rangle, \\ |v_i^\perp\rangle &= \frac{1}{\sqrt{1-q_i}} (1 - Q_i) |\psi\rangle. \end{aligned}$$

Note that we can write

$$|\psi\rangle = \sqrt{q_i}|v_i\rangle + \sqrt{1-q_i}|v_i^\perp\rangle. \tag{3.5}$$

We also define the state

$$|\psi_i^\perp\rangle = -\sqrt{1-q_i}|v_i\rangle + \sqrt{q_i}|v_i^\perp\rangle. \tag{3.6}$$

We can then use the above expressions to write $|v_i\rangle$ and $|v_i^\perp\rangle$ in terms of $|\psi\rangle$ and $|\psi_i^\perp\rangle$

$$\begin{aligned} |v_i\rangle &= \sqrt{q_i}|\psi\rangle - \sqrt{1-q_i}|\psi_i^\perp\rangle \\ |v_i^\perp\rangle &= \sqrt{1-q_i}|\psi\rangle + \sqrt{q_i}|\psi_i^\perp\rangle. \end{aligned} \tag{3.7}$$

The principal angle $\theta_i \in [0, \frac{\pi}{2}]$ between the two bases $\{|\psi\rangle, |\psi_i^\perp\rangle\}$ and $\{|v_i\rangle, |v_i^\perp\rangle\}$ is defined by

$$\cos^2\theta_i = |\langle v_i|\psi\rangle|^2 = \langle\psi|v_i\rangle\langle v_i|\psi\rangle = \langle\psi|Q_i|\psi\rangle = q_i. \tag{3.8}$$

Having defined the two bases $\{|v_i\rangle, |v_i^\perp\rangle\}$ and $\{|\psi\rangle, |\psi_i^\perp\rangle\}$, we are now ready to describe two algorithms for computing the expectation value q_i more efficiently than by using quantum state restoration. For any chosen ε and δ, each of these algorithms will generate an estimate q_i^{est} such that $|q_i^{\text{est}} - q_i| < \delta$ with probability at least $1 - \frac{\varepsilon}{d}$. Repeating for each i, we have $|q_i^{\text{est}} - q_i| < \delta$ for all i with probability at least $1 - \varepsilon$ by a union bound. The running times of these algorithms as a function of δ and ε are summarized in Table 3.1.

3.4.2.1. Alternating projections

This algorithm is an application of the scheme of Marriott and Watrous [24] which was originally proposed for witness-reusing amplification of the complexity class QMA. Observe from (3.5), (3.6) and (3.7) that when performing the measurement P on the state $|v_i\rangle$, the probability of obtaining 1 (and the state $|\psi\rangle$) is q_i. Similarly, when measuring P on the state $|v_i^\perp\rangle$, the probability of obtaining 0 (and the state $|\psi_i^\perp\rangle$) is also q_i. We can estimate q_i by performing many alternating measurements of P and Q_i and counting the number of transitions $|v_i\rangle \leftrightarrow |\psi\rangle$ or $|v_i^\perp\rangle \leftrightarrow |\psi_i^\perp\rangle$. Let us now present the algorithm and compute its complexity measured by the expected number of measurements of P, as a function of the desired precision δ and error probability $\frac{\varepsilon}{d}$.

1. Start with the state $|\psi\rangle$. Fix $N = \left\lceil \frac{1}{2} + \frac{\ln\frac{2d}{\varepsilon}}{4\delta^2} \right\rceil$.
2. Repeat for $t = 1, \ldots, N$
   a) Measure Q_i and record the measurement outcome as a bit $a_{2t-1} \in \{0, 1\}$. This produces one of the two states $|v_i\rangle$ or $|v_i^\perp\rangle$.
   b) Measure the projector $P = |\psi\rangle\langle\psi|$ and record the result $a_{2t} \in \{0, 1\}$. This produces either the state $|\psi\rangle$ or $|\psi_i^\perp\rangle$.
3. If the state is not currently $|\psi\rangle$ (because the last measurement in step 2b gave a 0), then the state is $|\psi_i^\perp\rangle$. In this case alternate measuring Q_i and P until you recover $|\psi\rangle$.
4. From the list $(a_1, \ldots, a_{2N})$, compute the list of differences
   $$(\Delta_1, \Delta_2, ..., \Delta_{2N-1}),$$
   where $\Delta_j = a_{j+1} \oplus a_j$. Let m denote the number of zeros in this list of differences. Then the estimate of q is given by
   $$q_i^{\text{est}} \equiv \frac{m}{2N-1}. \tag{3.9}$$

As discussed above, the probability of getting a measurement outcome (1 or 0) which is the same as the previous measurement outcome is q_i. So the number of zeros which appear in the list $(\Delta_1, \Delta_2, ..., \Delta_{2N-1})$ is a binomial random variable with mean $q_i(2N-1)$. This is why (3.9) gives an estimator for the value of q_i.

We now show that the estimate q_i^{est} from (3.9) has the required precision δ, with probability at least $1-\varepsilon$. To show this, we again use Hoeffding's inequality (3.4). Applying this to the case at hand with $q_k = 1 \oplus \Delta_k$ for $k \in \{1..., 2N-1\}$, we obtain

$$\Pr\left[|q_i^{\text{est}} - q_i| \geq \delta\right] \leq 2e^{-2(2N-1)\delta^2}.$$

The choice $N = \left\lceil \frac{1}{2} + \frac{\log \frac{2d}{\varepsilon}}{4\delta^2} \right\rceil$ guarantees that the right hand side is $\leq \frac{\varepsilon}{d}$. Thus we have shown that the desired precision δ is achieved by our scheme with probability at least $1 - \frac{\varepsilon}{d}$.

We now derive the expected number $\mathbb{E}[M_{\text{AP}}^{(i)}]$ (AP stands for alternating projections) of uses of P in the above algorithm. The random variable $M_{\text{AP}}^{(i)}$ is N plus the number of additional uses of P in step 3. The operation composed of measuring Q_i and then measuring P is an update of a symmetric random walk on the two states $\left\{|\psi\rangle, |\psi_i^{\perp}\rangle\right\}$. Let $w(r)$ be the probability of transitioning from $|\psi\rangle$ to $|\psi_i^{\perp}\rangle$ in r steps. Then with probability $1 - w(N)$ step 3 does not use P at all and, with probability $w(N)$ it uses an expected number $\frac{1}{w(1)}$ invocations of P. Thus the expected running time of the algorithm is

$$\begin{aligned}\mathbb{E}\left[M_{\text{AP}}^{(i)}\right] &= N + w(N)\frac{1}{w(1)} \\ &\leq 2N.\end{aligned}$$

In the last line, we used the fact that $w(N)$ is less than or equal to the probability of at least one transition occurring in N steps, which is at most $Nw(1)$ by a union bound.

Hence

$$\mathbb{E}[M_{\text{AP}}^{(i)}] \leq 2\left(\left\lceil \frac{1}{2} + \frac{1}{4\delta^2}\ln\frac{2d}{\varepsilon}\right\rceil\right).$$

Repeating this procedure to obtain estimates of each q_i (which are all within the desired precision δ with probability at least $1-\varepsilon$) takes the expected running time

$$\mathbb{E}[M_{\text{AP}}] \leq 2d\left(\left\lceil \frac{1}{2} + \frac{1}{4\delta^2}\ln\frac{2d}{\varepsilon}\right\rceil\right).$$

3.4.2.2. Phase estimation

In this section we will give an improved algorithm for single-copy tomography using phase estimation, based on a fast QMA amplification scheme given in [25].

Its advantage over the previous two algorithms is that it requires quadratically fewer measurements of P . The results of this section will prove Theorem 2.

As in the previous section, we estimate the q_i one at a time for $i \in \{1, \ldots, d\}$. We begin by defining the unitary operator

$$W_i = (2P - \mathbb{I})(2Q_i - \mathbb{I}),$$

which is a product of two reflections. Note that if we can implement P so that it coherently xors its measurement outcome into an ancilla register (as in the assumption of theorem 2), then we can implement the operator $(2P - \mathbb{I})$ by first initializing that ancilla to $|-\rangle$ and applying the measurement.

Within the 2D subspace S_i spanned by the vectors $|\psi\rangle$ and $|\psi_i^{\perp}\rangle$ (3.7), the operator W_i is a rotation

$$W_i\big|_{S_i} = e^{-2i\theta_i \sigma_y}, \tag{3.10}$$

where θ_i is the principal angle as defined in (3.8), and σ_y refers to the Pauli matrix.

We now describe how to obtain $q_i = \langle\psi|Q_i|\psi\rangle = \cos^2\theta_i$ by running phase estimation of the operator W_i on the state $|\psi\rangle$. The eigenvectors of W_i are

$$|\varphi_i^{\pm}\rangle = \frac{1}{\sqrt{2}}\left(|\psi\rangle \pm i|\psi_i^{\perp}\rangle\right). \tag{3.11}$$

and correspond to eigenvalues $e^{\mp i 2\pi\varphi_i}$, where $\varphi_i = \frac{\theta_i}{\pi}$ so that $0 < \varphi_i < \frac{1}{2}$. After running phase estimation of W_i on the input state $|\psi\rangle$, we will likely measure a good approximation to either φ_i or $1 - \varphi_i$. Note that either outcome provides a good estimate of

$$q_i = \cos^2(\pi\varphi_i) = \cos^2(\pi(1 - \varphi_i)).$$

This is the idea of the algorithm we present in this section. Our algorithm must have a failure probability lower than that obtained by a single use of phase estimation, and we must recover the state $|\psi\rangle$ at the end of the algorithm.

Our algorithm begins by defining

$$t = \left\lceil \log_2\left(\frac{3\pi}{\delta}\right)\right\rceil + 2.$$
$$r = \left\lceil \frac{1}{\log_2\left(\frac{2}{\sqrt{3}}\right)} \log_2\left(\frac{d}{2\varepsilon}\right)\right\rceil. \tag{3.12}$$

We proceed as follows:

1. Start in the state $|\psi\rangle|0\rangle^{\otimes t}$.
2. Repeat for $j = 1, ..., r$:

   a) Reset the t qubits of the second register to the state $|0\rangle^{\otimes t}$. Perform phase estimation of the operator W_i on the state of the first register, computing the phase using the t ancillas in the second register. Define

$$q_i^{(j)} = \cos^2\left(\pi\varphi_i^{(j)}\right)$$

   where $\varphi_i^{(j)}$ is the measured phase.

   b) Measure the projector $P = |\psi\rangle\langle\psi|$ on the first register.

3. If the state is not currently $|\psi\rangle$ (because the last measurement in step 2(b) gave a 0), then the state is $|\psi_i^\perp\rangle$. In this case repeat phase estimation followed by measurement of P until you measure a 1 for P, recovering the state $|\psi\rangle$.

4. Let q_i^{est} be the median of the values $\{q_i^{(j)}\}$ for $j \in \{1, ...r\}$.

We now determine the expected running time of this algorithm, and then we will show that the resulting estimate q_i^{est} achieves the desired precision with high enough probability. Our analysis of the running time is based on the observation that each iteration of phase estimation followed by measurement of P is an update of a random walk on the two states $\{|\psi\rangle, |\psi_i^\perp\rangle\}$. If we start in state $|\psi\rangle$ of the first register then after applying phase estimation (but before measuring the phase) we obtain a state

$$|\Psi_i\rangle = \frac{1}{\sqrt{2}}\left(|\varphi_i^+\rangle|\gamma\rangle + |\varphi_i^-\rangle|\mu\rangle\right).$$

where $|\gamma\rangle$ and $|\mu\rangle$ are t-qubit states. So the probability of measuring 1 in step 2b is

$$\Pr\left[|\psi\rangle \to |\psi\rangle\right] = \mathrm{Tr}\left[\left(|\psi\rangle\langle\psi| \otimes \mathbb{I}\right)|\Psi_i\rangle\langle\Psi_i|\right]$$

in which case the resulting state of the first register is $|\psi\rangle$. The probability of measuring a zero in this step is

$$\Pr\left[|\psi\rangle \to |\psi^\perp\rangle\right] = \mathrm{Tr}\left[\left(|\psi^\perp\rangle\langle\psi^\perp| \otimes \mathbb{I}\right)|\Psi_i\rangle\langle\Psi_i|\right] = 1 - \Pr\left[|\psi\rangle \to |\psi\rangle\right]$$

in which case the resulting state of the first register is $|\psi^\perp\rangle$. Similarly, one can compute the transition probabilities starting from the state $|\psi^\perp\rangle$ of the first register. These satisfy

$$\Pr\left[|\psi^\perp\rangle \to |\psi^\perp\rangle\right] = \Pr\left[|\psi\rangle \to |\psi\rangle\right]$$
$$\Pr\left[|\psi^\perp\rangle \to |\psi\rangle\right] = \Pr\left[|\psi\rangle \to |\psi^\perp\rangle\right]$$

so the random walk is symmetric. We can then directly apply our analysis of the previous section to show that

$$\mathbb{E}[\#\text{ of uses of phase estimation followed by measurement of } P] \leq 2r.$$

Each time we use phase estimation with t ancillas, we use the gate W_i less than 2^t times [28]. So each time we repeat phase estimation followed by measurement of P we use less than $2^t + 1$ measurements of P so the expected total number of times $\mathbb{E}[M_{\mathrm{PE}}^{(i)}]$ (PE stands for phase estimation) that we use the measurement of P is

$$\begin{aligned}\mathbb{E}[M_{\mathrm{PE}}^{(i)}] &< 2r \cdot \left(2^t + 1\right) \\ &\leq 2r \left(\frac{12\pi}{\delta} + 1\right) \\ &= 2 \left\lceil \frac{1}{\log_2\left(\frac{2}{\sqrt{3}}\right)} \log_2\left(\frac{d}{2\varepsilon}\right) \right\rceil \left(\frac{12\pi}{\delta} + 1\right).\end{aligned}$$

Repeating this procedure to obtain estimates of each q_i takes expected running time

$$\mathbb{E}[M_{\mathrm{PE}}] < 2d \left\lceil \frac{\log_2\left(\frac{d}{2\varepsilon}\right)}{\log_2\left(\frac{2}{\sqrt{3}}\right)} \right\rceil \left(\frac{12\pi}{\delta} + 1\right). \tag{3.13}$$

We now show that the probability that all the estimates q_i^{est} obtained by using the above algorithm satisfy

$$|q_i^{est} - q_i| < \delta$$

is at least $1 - \varepsilon$. Our choice of t was designed so that the output of phase estimation of W_i on the state $|\varphi_i^+\rangle$ using t ancillas is a state $|\varphi_i^+\rangle|\gamma\rangle$ such that a measurement of the t-qubit state $|\gamma\rangle$ in the computational basis produces a phase $\tilde{\varphi}$ that satisfies

$$|\tilde{\varphi} - \varphi_i| \leq \frac{\delta}{3\pi}$$

with probability at least ¾ [28]. Similarly the output of phase estimation of W_i on the state $|\varphi_i^-\rangle$ using t ancillas is a state $|\varphi_i^-\rangle|\mu\rangle$ such that a measurement of the t-qubit state $|\mu\rangle$ in the computational basis produces a phase $\tilde{\varphi}$ that satisfies

$$|\tilde{\varphi} - (1 - \varphi_i)| \leq \frac{\delta}{3\pi}$$

with probability at least ¾. In step 2(a) of our algorithm we perform phase estimation on either the state $|\psi\rangle$ or the state $|\psi_i^\perp\rangle$. In either case, the reduced density matrix of the t-qubit ancilla register after applying the phase estimation (but before measuring the phase) is

$$\frac{1}{2}\left(|\gamma\rangle\langle\gamma| + |\mu\rangle\langle\mu|\right)$$

which is an equal probabilistic mixture of $|\gamma\rangle$ and $|\mu\rangle$. So, with probability at least ¾ (regardless of whether we started in $|\psi\rangle$ or $|\psi^\perp\rangle$), the phases $\varphi_i^{(j)}$ measured in

State Restoration	Alternating Projectors	Phase Estimation
$\mathbb{E}[M_{\mathrm{SR}}] = \mathrm{O}\left(\frac{\chi \cdot d}{\delta^2} \log \frac{d}{\varepsilon}\right)$	$\mathbb{E}[M_{\mathrm{AP}}] = O\left(\frac{d}{\delta^2} \log \frac{d}{\varepsilon}\right)$	$\mathbb{E}[M_{\mathrm{PE}}] = O\left(\frac{d}{\delta} \log\left(\frac{d}{\varepsilon}\right)\right)$

Table 3.1.: Scaling of the expected number of measurements of $P = |\psi\rangle\langle\psi|$ used by each algorithm as a function of the desired precision δ and error probability ε.

step 2 of the algorithm satisfy either

$$|\varphi_i^{(j)} - \varphi_i| \leq \frac{\delta}{3\pi}$$

or

$$|\varphi_i^{(j)} - (1 - \varphi_i)| \leq \frac{\delta}{3\pi}.$$

Using the inequality

$$|\cos^2(\pi\alpha) - \cos^2(\pi\beta)| \leq 2\pi|\alpha - \beta|$$

and the fact that $\cos^2(\pi x) = \cos^2(\pi(1-x))$ it follows that the estimates $q_i^{(j)}$ each (independently) satisfy

$$|q_i^{(j)} - q_i| < \delta$$

with probability at least ¾. The median lemma of [25] says in this case that the probability that the median of the r independent measured values $q_i^{(j)}$ falls outside the interval $(q_i - \delta, q_i + \delta)$ is upper bounded as $p_{\mathrm{fail}} \leq \frac{1}{2}\left(\frac{\sqrt{3}}{2}\right)^r$. Plugging in our choice of r from equation (3.12) gives

$$|q_i^{\mathrm{est}} - q_i| < \delta$$

for each i with probability at least $1 - \frac{\varepsilon}{d}$. So the probability that the above inequality is satisfied for all of the $i \in \{1, \ldots d\}$ is at least $1 - \varepsilon$.

3.4.3. Performance comparison

These three algorithms for estimating the probabilities $q_i = \mathrm{Tr}\left[\rho_B |i\rangle_B \langle i|_B\right]$ give estimates $\{q_i^{\mathrm{est}}\}$ (for i from 1 to d) which are all within δ of the correct values with probability at least $1 - \varepsilon$. Their running times are summarized in table 3.1.

State restoration is conceptually the simplest of the three algorithms, and we expect that it will be sufficient for most purposes. It is also the slowest as a function of d and δ (assuming χ is increasing as a function of d). The state restoration algorithm has the advantage that we can drop in different tomography schemes that may improve performance.

In the absence of a better tomography scheme, however, both other algorithms outperform state restoration as a function of d. Phase estimation also performs quadratically better than both other algorithms as $\delta \to 0$.

3.5. A condensed-matter application of single-copy tomography

Quantum computers offer potentially exponential speedups in simulating quantum mechanics, but some problems are still hard. For example, preparing ground states of many-body systems generically takes exponential time in the number of particles. Nonetheless, for sufficiently small systems with large enough energy gaps, algorithms such as [29] may run quickly enough to prepare a single copy of the ground state, and phase estimation can be used to verify the ground state. Single-copy tomography allows us to make multiple tomographic measurements (even of non-commuting operators) on small numbers of particles without having to prepare multiple copies of the ground state. This gives a large speedup over traditional tomography.

Single-copy tomography could also be useful to characterize the ground state during adiabatic evolution. This information could even be used in real time to guide the choice of path for an adiabatic algorithm.

3.6. A faster attack against Wiesner's money

In the beginning of this chapter, we described a simple attack against any public-key quantum money if the quantum state is a product state. Wiesner's money is vulnerable to a faster attack.

In Wiesner's money, each qubit $|\psi_i\rangle$ is either $|0\rangle$, $|1\rangle$, $|+\rangle$, or $|-\rangle$. All of these states are eigenstates of either X or Z.

To counterfeit the quantum money, an attacker named Carla can learn the full description of a quantum bill by using a verification machine exactly once per qubit. Suppose that Carla's quantum money state is $|\$\rangle$. To learn the i^{th} qubit, she tries to verify the state $X_i|\$\rangle$. If $X_i|\$\rangle$ is invalid, then the state $|\psi_i\rangle$ is either $|0\rangle$ or $|1\rangle$, as the other possibilities $|+\rangle$ and $|-\rangle$ are eigenstates of X_i. In this case, the verification machine does not change the state it received, Carla now knows that $|\psi_i\rangle$ is an eigenstate of Z. She applies X_i to recover $|\$\rangle$ and measures $|\psi_i\rangle$ in the Z basis to learn whether it is $|0\rangle$ or $|1\rangle$.

If, on the other hand, the state $X_i|\$\rangle$ is valid, then $|\psi_i\rangle$ is either $|-\rangle$ or $|+\rangle$. In this case, the verification machine gives Carla her state $X_i|\$\rangle = |\$\rangle$ back. But now Carla knows that $|\psi_i\rangle$ is an eigenstate of X and she can measure it to learn whether it is $|+\rangle$ or $|-\rangle$.

If Carla repeats this process for $i = 1, \ldots, n$, she will learn the secret description of $|\$_s\rangle$ in exactly n queries to the verifier. Once she has done this, she can make as many counterfeit copies of $|\$_s\rangle$ as she wants.

If Wiesner's quantum money is used as a private-key quantum money scheme, the mint must be careful to prevent this kind of attack. If the mint returns invalid bills back to their owners as souvenirs, then Carla could perform the exact same attack using the mint as a verification machine. The mint must destroy any invalid

bill it sees to prevent this attack.

4. Breaking Aaronson's quantum money

In 2009, Aaronson proposed [20] a public-key quantum money scheme. His quantum money is a stabilizer state. To create a quantum bill, the mint generates a random stabilizer state with certain properties and constructs an obfuscated description of the state that can be used to verify that state. The quantum bill is the quantum state, the obfuscated description used for verification, and a digital signature of the verifier. The scheme is parametrized by integers n, m and l and by a real number $\varepsilon \in [0, 1]$. These parameters are required to satisfy $\frac{1}{\varepsilon^2} \ll l$.

Stabilizer money is insecure due to two different attacks. If ε is small, then the verification procedure is too weak and a counterfeiter can construct a different quantum state that will appear valid. If ε is large, then the description of the quantum state can be deobfuscated and a counterfeiter can produce new quantum money states directly. These attacks require only the obfuscated description of a valid money state – neither attack requires access to a valid quantum money state.

The quantum money state is a tensor product of l different stabilizer states, each on n qubits, and the classical secret is a list of Pauli group operators which stabilize the state. The bank generates an instance of the money by choosing a random stabilizer state for each of the l registers. To produce the verifier, the bank generates an $m \times l$ table of n qubit Pauli group operators. The $(i, j)^{\text{th}}$ element of the table is an operator

$$E_{ij} = (-1)^{b_{ij}} A_1^{ij} \otimes A_2^{ij} ... \otimes A_n^{ij}$$

where each $A_k^{ij} \in \{1, \sigma_x, \sigma_y, \sigma_z\}$ and $b_{ij} \in \{0, 1\}$. Each element E_{ij} of the table is generated by the following procedure:

1. With probability $1 - \varepsilon$ choose the b_{ij} and, for each k, A_k^{ij} uniformly at random.

2. With probability ε choose the operator E_{ij} to be a uniformly random element of the stabilizer group of $|C_i\rangle$.

To verify the quantum money state, for each i the authenticator chooses $j\,(i) \in [m]$ at random and measures

$$Q = \frac{1}{l} \sum_i I^{\otimes i-1} \otimes E_{i,j(i)} \otimes I^{\otimes m-i}. \tag{4.1}$$

The authenticator accepts iff the outcome is greater than or equal to $\frac{\varepsilon}{2}$. Note that measuring the operator Q is equivalent to measuring the operator $E_{i,j(i)}$ for each

register $i \in [l]$ and then averaging the results, since the measurements on different registers commute.

The state $|C_1\rangle|C_2\rangle...|C_l\rangle$ is accepted by this procedure with high probability since the probability of measuring a $+1$ for the operator $E_{i,j(i)}$ on the state $|C_i\rangle$ is $\frac{1+\varepsilon}{2}$. The mean value of the operator Q in the state $|C_1\rangle|C_2\rangle...|C_l\rangle$ is therefore ε, since it is simply the average of the $E_{i,j(i)}$ for each register $i \in [l]$. The parameter l is chosen so that $\frac{l}{\varepsilon^2} = \Omega(n)$ so the probability that one measures Q to be less than $\frac{\varepsilon}{2}$ is exponentially small in n.

4.1. The attack

Our attack on this money depends on the parameter ε. Our proofs assume that $m = \text{poly}(n)$, but we expect that both attacks work beyond the range in which our proofs apply.

4.1.1. Attacking the verifier for small ϵ

For $\varepsilon \leq \frac{1}{16\sqrt{m}}$ and with high probability over the table of Pauli operators, we can efficiently generate a state that passes verification with high probability. This is because the verification algorithm does not project onto the intended money state but in fact accepts many states with varying probabilities. On each register, we want to produce a state for which the expected value of the measurement of a random operator from the appropriate column of E is sufficiently positive. This is to ensure that, with high probability, the verifier's measurement of Q will have an outcome greater than $\frac{\varepsilon}{2}$. For small ε, there are many such states on each register and we can find enough of them by brute force.

We find states that pass verification by working on one register at a time. For each register i, we search for a state ρ_i with the property that

$$\text{Tr}\left[\left(\frac{1}{m}\sum_{j=1}^{m} E_{ij}\right)\rho_i\right] \geq \frac{1}{4\sqrt{m}} + O\left(\frac{1}{m^2}\right). \tag{4.2}$$

As we show in Section 4.2, we can find such states efficiently on enough of the registers to construct a state that passes verification.

4.1.2. Recovering the classical secret for large ϵ

For $\varepsilon \geq \frac{c}{\sqrt{m}}$ for any constant $c > 0$, we use a classical attack instead: we recover the classical secret (i.e. a description of the quantum state) and can thus forge the money. We observe that each column of the table E contains approximately εm commuting operators, with the rest chosen randomly, and if, in each column, we can find a set of commuting operators that is at least as large as the planted set, then any quantum state stabilized by these operators will pass verification.

We begin by casting our question as a graph problem. For each column, let G be a graph whose vertices correspond to the m measurements, and connect vertices i and j if and only if the corresponding measurements commute. The vertices corresponding to the planted commuting measurements now form a clique, and we aim to find it.

In general, it is intractable to find the largest clique in a graph. In fact, it is NP-hard even to approximate the size of the largest clique within $n^{1-\varepsilon}$, for any $\varepsilon > 0$ [30]. Finding large cliques planted in otherwise random graphs, however, can be easy.

For example, if $\varepsilon = \Omega\left(\frac{\log m}{\sqrt{m}}\right)$, then a simple classical algorithm will find the clique. This algorithm proceeds by sorting the vertices in decreasing order of degree and selecting vertices from the beginning of the list as long as the selected vertices continue to form a clique.

We can find the planted clique for $\varepsilon \geq \frac{c}{\sqrt{m}}$ for any constant $c > 0$ in polynomial time using a more sophisticated classical algorithm that may be of independent interest. If the graph were obtained by planting a clique of size $\varepsilon\sqrt{m}$ in a random graph drawn from $G(m, 1/2)$, Alon, Krivelevich, and Sudakov showed in [31] that one can find the clique in polynomial time with high probability.[1] Unfortunately, the measurement graph G is not drawn from $G(m, 1/2)$, so we cannot directly apply their result. However, we show in section 4.2 that if G is sufficiently random then a modified version of their algorithm works.

4.2. Details of the attack against stabilizer money for small ϵ

For $\varepsilon \leq \frac{1}{16\sqrt{m}}$ and with high probability in the table of Pauli operators, we can efficiently generate a state that passes verification with high probability. Our attack may fail for some choices of the table used in verification, but the probability that such a table of operators is selected by the bank is exponentially small.

Recall that each instance of stabilizer money is verified using a classical certificate, which consists of an $m \times l$ table of n qubit Pauli group operators. The $(i, j)^{\text{th}}$ element of the table is an operator

$$E_{ij} = (-1)^{b_{ij}} A_1^{ij} \otimes A_2^{ij} \ldots \otimes A_n^{ij}$$

where each $A_k^{ij} \in \{1, \sigma_x, \sigma_y, \sigma_z\}$ and $b_{ij} \in \{0, 1\}$.

We will use one important property of the algorithm that generates the table of Pauli operators: with the exception of the fact that $-I^{\otimes n}$ cannot occur in the table, the distribution of the tables is symmetric under negation of all of the operators.

[1] Remember that $G(m, p)$ is the Erdös-Rényi distribution over m-vertex graphs in which an edge connects each pair of vertices independently with probability p. The AKS algorithm was later improved [32] to work on subgraphs of $G(n, p)$ for any constant p, but our measurement graph G is not of that form.

The verification algorithm works by choosing, for each i, a random $j\,(i) \in [m]$. The verifier then measures

$$Q = \frac{1}{l}\sum_i I^{\otimes i-1} \otimes E_{i,j(i)} \otimes I^{\otimes m-i}. \tag{4.3}$$

The algorithm accepts iff the outcome is greater than or equal to $\frac{\varepsilon}{2}$. Note that measuring the operator Q is equivalent to measuring the operator $E_{i,j(i)}$ for each register $i \in [l]$ and then averaging the results, since the measurements on different registers commute.

To better understand the statistics of the operator Q, we consider measuring an operator $E_{i,j(i)}$ on a state ρ_i, where $j(i) \in [m]$ is chosen uniformly at random. The total probability $p_1(\rho_i)$ of obtaining the outcome $+1$ is given by

$$\begin{aligned} p_1(\rho_i) &= \frac{1}{m}\sum_{j=1}^{m} \mathrm{Tr}\left[\left(\frac{1+E_{i,j(i)}}{2}\right)\rho_i\right] \\ &= \frac{1+\mathrm{Tr}\left[H^{(i)}\rho_i\right]}{2} \end{aligned}$$

where (for each $i \in [l]$) we have defined the Hamiltonian

$$H^{(i)} = \frac{1}{m}\sum_{j=1}^{m} E_{ij}.$$

We use the algorithm described below to independently generate an n qubit mixed state ρ_i on each register $i \in [l]$. At least $1/4$ of these states ρ_i (w.h.p. over the choice of the table E) will have the property that

$$\mathrm{Tr}[H^{(i)}\rho_i] \geq \frac{1}{4\sqrt{m}} + O\left(\frac{1}{m^2}\right) \tag{4.4}$$

and the rest have

$$p_1(\rho_i) \geq \frac{1}{2} - O\left(\frac{1}{m}\right) \tag{4.5}$$

which implies that

$$\mathbb{E}_i\, p_1(\rho_i) \geq \frac{1}{2} + \frac{1}{8\sqrt{m}} + O\left(\frac{1}{m^2}\right).$$

We use the state

$$\rho = \rho_1 \otimes \rho_2 \otimes \ldots \otimes \rho_l$$

as our forged quantum money. If the verifier selects $j\,(i)$ at random and measures Q (from equation (4.3)), then the expected outcome is at least $\frac{1}{4}(\frac{1}{4\sqrt{m}} + O(\frac{1}{m^2})) + \frac{3}{4}O(\frac{1}{m})$, and the probability of an outcome less than $\frac{1}{32\sqrt{m}}$ (for $\varepsilon \leq \frac{1}{16\sqrt{m}}$, the

verifier can only reject if this occurs) is exponentially small for m sufficiently large by independence of the registers. Therefore the forged money state ρ is accepted by Aaronson's verifier with probability that is exponentially close to 1 if $\varepsilon \leq \frac{1}{16\sqrt{m}}$.

Before describing our algorithm to generate the states $\{\rho_i\}$, we must understand the statistics (in particular, the first two moments) of each $H^{(i)}$ on the fully mixed state $\frac{\mathbb{I}}{2^n}$. We will assume that, for $j \neq k$, $E_{ij} \neq E_{ik}$. We also assume that the operators $\pm I \otimes I \otimes I ... \otimes I$ do not appear in the list. Both of these assumptions are satisfied with overwhelming probability. The first and second moments of $H^{(i)}$ are

$$\mathrm{Tr}\left[H^{(i)}\frac{\mathbb{I}}{2^n}\right] = 0$$

and

$$\mathrm{Tr}\left[\left(H^{(i)}\right)^2\frac{\mathbb{I}}{2^n}\right] \tag{4.6}$$

$$= 2^{-n}\,\mathrm{Tr}\left[\frac{1}{m^2}\sum_j (E_{i,j})^2 + \frac{1}{m^2}\sum_{j\neq k} E_{i,j}E_{i,k}\right]$$

$$= \frac{1}{m}. \tag{4.7}$$

We define f_i to be the fraction (out of 2^n) of the eigenstates of $H^{(i)}$ that have eigenvalues in the set $[\frac{1}{2\sqrt{m}}, 1] \cup [-1, -\frac{1}{2\sqrt{m}}]$. Since the eigenvalues of $H^{(i)}$ are bounded between -1 and 1, we have

$$\mathrm{Tr}\left[\left(H^{(i)}\right)^2\frac{\mathbb{I}}{2^n}\right] \leq f_i + (1-f_i)\frac{1}{4m}.$$

Plugging in equation (4.7) and rearranging we obtain

$$f_i \geq \frac{3}{4m-1}.$$

We also define g_i to be the fraction of eigenstates of $H^{(i)}$ that have eigenvalues in the set $[\frac{1}{2\sqrt{m}}, 1]$. The distribution (for any fixed i) of E_{ij} as generated by the bank is symmetric under negation of all the E_{ij}, so with probability at least ½ over the choice of the operators in the row labeled by i, the fraction g_i satisfies

$$g_i \geq \frac{3}{8m-2}. \tag{4.8}$$

We assume this last inequality is satisfied for at least ¼ of the indices $i \in [l]$, for the particular table E_{ij} that we are given. The probability that this is not the case is exponentially small in l.

Ideally, we would generate the states ρ_i by preparing the fully mixed state, measuring $H^{(i)}$, keeping the result if the eigenvalue is at least $\frac{1}{2\sqrt{m}}$, and otherwise trying again, up to some appropriate maximum number of tries. After enough failures, we would simply return the fully mixed state. It is easy to see that outputs of this algorithm would satisfy equation (4.2) with high probability.

Unfortunately, we cannot efficiently measure the exact eigenvalue of an arbitrary Hermitian operator; instead, we use phase estimation, which gives polynomial error using polynomial resources. In section 4.2.2 we review the phase estimation algorithm. In section 4.2.1, we describe an efficient algorithm to generate ρ_i using phase estimation and show that the resulting states, even in the presence of errors due to polynomial-time phase estimation, are accepted by the verifier with high probability, assuming that the table E_{ij} has the appropriate properties.

4.2.1. Generating each register's state

We now fix a particular value of i and, for convenience, define $H = \frac{1}{4}H^{(i)}$ so that all the eigenvalues of H lie in the interval $[-\frac{1}{4}, \frac{1}{4}]$. We denote the eigenvectors of H by $\{|\psi_j\rangle\}$ and write

$$e^{2\pi i H}|\psi_j\rangle = e^{2\pi i \varphi_j}|\psi_j\rangle.$$

The positive eigenvalues of H map to phases φ_j in the range$[0, \frac{1}{4}]$ and negative eigenvalues of H map to $[\frac{3}{4}, 1]$.

We label each eigenstate of H as either "good" or "bad" according to its energy. We say an eigenstate $|\psi_j\rangle$ is good if $\varphi_j \in [\frac{1}{16\sqrt{m}}, \frac{1}{4}]$. Otherwise we say it is bad (which corresponds to the case where $\varphi_j \in [0, \frac{1}{16\sqrt{m}}) \cup [\frac{3}{4}, 1]$.

We use the following algorithm to produce a mixed state ρ_i.

1. Set $k = 1$.

2. Prepare the completely mixed state $\frac{\mathbb{I}}{2^n}$. In our analysis of this step, we will imagine that we have selected an eigenstate $|\psi_p\rangle$ of H uniformly at random, which yields identical statistics.

3. Use the phase estimation circuit to measure the phase of the operator $e^{2\pi i H}$. Here the phase estimation circuit (see section 4.2.2) acts on the original n qubits in addition to $q = r + \lceil \log(2 + \frac{2}{\delta}) \rceil$ ancilla qubits, where we choose

$$r = \lceil \log(20m) \rceil$$
$$\delta = \frac{1}{m^3}.$$

4. Accept the resulting state (of the n qubit register) if the measured phase $\varphi' = \frac{z}{2^q}$ is in the interval $[\frac{1}{8\sqrt{m}} - \frac{1}{20m}, \frac{1}{2}]$. In this case stop and output the state of the first register. Otherwise set $k = k + 1$.

5. If $k = m^2 + 1$ then stop and output the fully mixed state. Otherwise go to step 2.

We have chosen the constants in steps 3 and 4 to obtain an upper bound on the probability p_b of accepting a bad state in a particular iteration of steps 2, 3, and 4:

$$\begin{aligned} p_b &= \Pr\left(|\psi_p\rangle \text{ is bad and you accept}\right) \\ &\leq \Pr\left(\text{accept given that } |\psi_p\rangle \text{ was bad}\right) \\ &\leq \Pr\left(|\varphi_p - \varphi'| > \frac{1}{16\sqrt{m}} - \frac{1}{20m}\right) \\ &\leq \Pr\left(|\varphi_p - \varphi'| > \frac{1}{20m}\right) \\ &\leq \delta \text{ by equation 4.13.} \end{aligned}$$

Above, we considered two cases depending on whether or not the inequality 4.8 is satisfied for the register i. We analyze the algorithm in these two cases separately.

Case 1: Register i satisfies inequality 4.8

In this case, choosing p uniformly,

$$\Pr\left(\frac{1}{4} \geq \varphi_p \geq \frac{1}{8\sqrt{m}}\right) \geq \frac{3}{8m-2} \tag{4.9}$$

This case occurs for at least ¼ of the indices $i \in [l]$ with all but exponential probability.

The probability p_g that you pick a good state (in a particular iteration of steps 2, 3, and 4) and then accept it is at least

$$\begin{aligned} p_g &= \Pr\left(|\psi_p\rangle \text{ is good and you accept}\right) \\ &\geq \Pr\left(\frac{1}{4} \geq \varphi_p \geq \frac{1}{8\sqrt{m}} \text{ and you accept}\right) \\ &= \Pr\left(\frac{1}{4} \geq \varphi_p \geq \frac{1}{8\sqrt{m}}\right) \\ &\qquad \times \Pr\left(\text{accept given } \frac{1}{4} \geq \varphi_p \geq \frac{1}{8\sqrt{m}}\right) \\ &\geq \Pr\left(\frac{1}{4} \geq \varphi_p \geq \frac{1}{8\sqrt{m}}\right)(1-\delta) \\ &\geq \frac{3}{8m-2}\left(1 - \frac{1}{m^3}\right) \\ &\geq \frac{1}{4m}\text{ , for m sufficiently large.} \end{aligned}$$

Thus the total probability of outputting a good state is (in a complete run of the algorithm)

$$\begin{aligned}
&\Pr(\text{output a good state}) && (4.10)\\
&= \sum_{k=1}^{m^2} p_g(1-p_g-p_b)^{k-1}\\
&= \frac{p_g}{p_g+p_b}\left(1-(1-p_g-p_b)^{m^2}\right)\\
&\geq \frac{p_g}{p_g+p_b}\left(1-(1-p_g)^{m^2}\right)\\
&\geq \frac{p_g}{p_g+\delta}\left(1-(1-p_g)^{m^2}\right).\\
&\geq \frac{p_g}{p_g+\delta}\left(1-e^{-p_g m^2}\right) && (4.11)\\
&\geq \frac{1}{1+\frac{4}{m^2}}\left(1-e^{-p_g m^2}\right) \text{ for m sufficiently large.}\\
&= 1-O\left(\frac{1}{m^2}\right)
\end{aligned}$$

So in this case, the state ρ_i will satisfy

$$\begin{aligned}
&\mathrm{Tr}\left[H^{(i)}\rho_i\right]\\
&\geq \Pr\left(\text{output a good state}\right)\frac{1}{4\sqrt{m}}\\
&\quad -\left(1-Pr\left(\text{output a good state}\right)\right)\\
&= \frac{1}{4\sqrt{m}}+O\left(\frac{1}{m^2}\right).
\end{aligned}$$

Case 2: Register i does not satisfy inequality 4.8

This case occurs for at most ¾ of the indices $i \in [l]$ with all but exponentially small probability.

The probability of accepting a bad state for register i at any point is

$$\Pr\left(\text{accept a bad state ever}\right) \leq \sum_{k=1}^{m^2}\delta = \frac{1}{m}. \quad (4.12)$$

So the state ρ_i which is generated by the above procedure will satisfy

$$\begin{aligned}
&\mathrm{Tr}\left[H^{(i)}\rho_i\right]\\
&\geq -\Pr\left(\text{accept a bad state ever}\right)\\
&= -\frac{1}{m}.
\end{aligned}$$

We have thus shown that equation (4.4) holds for all indices i that satisfy inequality 4.8; equation (4.5) holds for the rest of the indices. As discussed above, this guarantees (assuming at least ¼ of the indices i satisfy inequality (4.8)) that our forged state $\rho = \rho_1 \otimes \rho_2 \otimes ... \otimes \rho_l$ is accepted by the verifier with high probability if $\varepsilon \leq \frac{1}{16\sqrt{m}}$.

4.2.2. Review of the phase estimation algorithm

In this section we review some properties of the phase estimation algorithm as described in [28]. We use this algorithm in section 4.2 to measure the eigenvalues of the operator $e^{2\pi i H}$. The phase estimation circuit takes as input an integer r and a parameter δ and uses

$$q = r + \lceil log(2 + \frac{2}{\delta}) \rceil$$

ancilla qubits. When used to measure the operator $e^{2\pi i H}$, phase estimation requires a subroutine that implements the unitary operator $e^{2\pi i H t}$ for $t \leq 2^r$; this can be approximated efficiently if $2^r = O(poly(n))$. This approximation of the Hamiltonian time evolution incurs an error that can be made polynomially small in n using polynomial resources (see for example [28]). We therefore neglect this error in the remainder of the discussion. The phase estimation circuit, when applied to an eigenstate $|\psi_j\rangle$ of H such that

$$e^{2\pi i H}|\psi_j\rangle = e^{2\pi i \varphi_j}|\psi_j\rangle,$$

and with the q ancillas initialized in the state $|0\rangle^{\otimes q}$, outputs a state

$$|\psi_j\rangle \otimes |a_j\rangle$$

where $|a_j\rangle$ is a state of the ancillas. If this ancilla register is then measured in the computational basis, the resulting q bit string z will be an approximation to φ_j that is accurate to r bits with probability at least $1 - \delta$ in the sense that

$$\Pr\left(\left|\varphi_j - \frac{z}{2^q}\right| > \frac{1}{2^r}\right) \leq \delta. \tag{4.13}$$

In order for this algorithm to be efficient, we choose r and δ so that $2^r = \text{poly}(n)$ and $\delta = \frac{1}{\text{poly}(n)}$.

5. Quantum money based on component mixers

Our approach to collision-free quantum money is to make the quantum money state $|\$_\ell\rangle$ easy to describe but hard to produce.

The description of our state starts with a very large classical set S. The set S is the computational basis of the Hilbert space that our money lives in. We partition S into a large number c of subsets $S_1, \ldots, S_c$. Each component has a label ℓ, and a classical function maps each element of S to the label of the component that it is in. The state $|\$_\ell\rangle$ is simply the uniform superposition

$$|\$_\ell\rangle = \frac{1}{\sqrt{N_\ell}} \sum_{\substack{x \in S \\ L(x)=\ell}} |x\rangle \tag{5.1}$$

of all of the elements in the component with label ℓ. The normalization factor N_ℓ is the size of the component with label ℓ.

The mint prepares quantum money states by postselection. To produce each money state, it starts by preparing the uniform superposition

$$\frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle$$

of everything.[1] The mint then coherently computes L into an ancilla register, giving

$$\frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle|L(x)\rangle.$$

Finally, the mint measures the ancilla register. This collapses the state to

$$|\$_\ell\rangle|\ell\rangle = \frac{1}{\sqrt{N_\ell}} \sum_{\substack{x \in S \\ L(x)=\ell}} |x\rangle|\ell\rangle = |\$_\ell\rangle|\ell\rangle.$$

The measurement result $|\ell\rangle$ is the serial number of the new quantum money state and the quantum output $|\$_\ell\rangle$ is the quantum money.

For this to be collision-free, we need the measurement ℓ to never repeat itself. This means that, for some security parameter n, there should be an exponentially

[1] This can be done in any number of ways. If S is encoded densely enough as strings of bits, then a Hadamard transform on $|0\rangle^{\otimes n}$ followed by checking that the result is in S will work. If not, the algorithm in [33] will usually work.

large set of labels and an exponentially large number of elements with each label. Furthermore, no label should correspond to more than an exponentially small fraction of the set.

The function L should be as obscure and have as little structure as possible. This makes the quantum money state $|\psi_\ell\rangle$ be the uniform superposition of exponentially many terms that seemingly have no useful relationship to each other. Since no label occurs during the postselection procedure above with greater than exponentially small probability, the postselection procedure would have to be repeated exponentially many times to produce the same label ℓ twice. If the labeling function L is a black box with no additional structure, then the Grover lower bound [34] rules out any polynomial time algorithm that can produce the state $|\psi_\ell\rangle$ given only knowledge of ℓ. We conjecture that it is similarly difficult to copy a state $|\psi_\ell\rangle$ or to produce the state $|\psi_\ell\rangle \otimes |\psi_\ell\rangle$ for any ℓ at all.

In order to verify a quantum money state, a merchant first measures L again on that state and confirms that the measurement result matches the alleged serial number of the state. The merchant must then verify that the state is the correct superposition. To make this possible, we need some additional structure in the components. We need a family of functions $\{M_i\}$ with the property that, if i is selected uniformly at random from the set of all possible values, then M_i will map an element of any component of S to a new, approximately uniformly random element of the *same* component of S. The merchant can verify the superposition by checking that the quantum state is invariant under the action of the functions M_i.

To encode the quantum money as qubits, we assume that S is a set of n-bit strings of ones and zeros. The number of bits n will be the security parameter of the quantum money.

The set of functions $\{M_i\}$ is called a component mixer because it mixes within components of S. In the remainder of this chapter, I give evidence that quantum money based on component mixers is secure.

5.1. Introduction

In a quantum money protocol based on component mixers, all parties (the mint, honest users of money, and any adversaries) have access to two black-box operations. They have access to a "component mixer" (defined below) that invertibly maps any string to a new almost uniformly random string in the same component. They also have access to a labeling function that determines which component any element is in. (In the concrete scheme, the component mixer does not fully mix within the components. We ignore that issue here.)

To prove hardness results related to counterfeiting quantum money, we need an appropriate computational assumption. We find this assumption in a new class of query problems based on component mixers.

All these problems involve a large set that is partitioned into components. An algorithm must use black-box queries to a component mixer to answer questions about the components. The algorithm is not given access to a labeling function. The SAME COMPONENT problem is: are two given elements in the same component? The MULTIPLE COMPONENTS problem is: is there more than one component (as opposed to just one)? If we promise that either there is only one component or no component contains a majority of the set, then the MULTIPLE COMPONENTS problem becomes MULTIPLE BALANCED COMPONENTS. Finally, on a quantum computer, the COMPONENT SUPERPOSITION problem is to prepare the uniform superposition of all elements in a component given as input one element in that component. (The classical analog, producing a uniformly random sample from a component, is easy by assumption.)

These types of questions are natural abstractions of graph isomorphism and group membership. Graph isomorphism is the problem of deciding whether two graphs are equivalent up to a permutation. The complexity of graph isomorphism is unknown on both classical and quantum computers. Group membership, on the other hand, is the problem of deciding whether an element of some large group is in a particular subgroup of that group. The subgroup is specified by its generators, and the group structure is given either as a black box or in some explicit form such as a matrix representation. In the black-box setting, group membership is hard on classical computers [35] but has unknown complexity on quantum computers.

For graph isomorphism, the big set would be the set of all graphs of a given size and the components are isomorphism classes of graphs. The component mixer permutes the vertices of a graph. Testing whether two graphs are isomorphic reduces to the SAME COMPONENT problem.

For group membership, the big set would be a large group and the components would be cosets of a subgroup that is described only by its generators. The component mixer would multiply by an element of the subgroup. The group membership problem reduces to an instance of SAME COMPONENT: testing whether a given element is in the same component as the identity. The MULTIPLE BALANCED COMPONENTS problem would determine whether the given generators generate the entire group or a proper subgroup.

Each of these query problems naturally defines a complexity class. SCP, MCP, and MBCP are the sets of languages that are polynomial-time reducible to the SAME COMPONENT problem, MULTIPLE COMPONENTS problem and MULTIPLE BALANCED COMPONENTS problem. We relate all three classes to commonly-used complexity classes. Our results are summarized in the table above.

These problems and classes are immediately interesting for two reasons. First, if SAME COMPONENT is hard on a quantum computer, then we have evidence for the security of a quantum money protocol [2]. Second, MCP and MBCP are candidates for a classical oracle separation between QCMA and QMA. (Group membership does not work directly because it has too much structure [36].)

The class...	...is in...	...and oracle-separated from
SCP	NP, SZK	co-MA, hopefully BQP
MCP	QMA, $NP^{co\text{-}NP}$, $NP^{co\text{-}SCP}$	BQP, hopefully QCMA
MBCP	MCP, AM, co-AM, SZK, BPP^{SCP}	hopefully QCMA

Table 5.1.: Component problems have a home in the complexity zoo.

5.2. Definitions

All the problems we consider are questions about a large set S. For consistency in defining the size of the problems, we take n to be the number of bits used to represent an element of S. The set S is partitioned into components, and access to the components is given through a family of invertible maps that takes each element of every component of S to a new element of the same component. The set of maps constitutes a component mixer if a uniform random choice of which map in the set to evaluate produces a uniformly random output.

Definition 3. A family of one-to-one maps $\{M_i\}$ is a *component mixer* on a partition $\{S_1, \ldots, S_c\}$ of a set S if:

- The set S is a subset of n-bit strings.
- The family is indexed by a label i from a set Ind_M, and each i can be encoded in $O(\text{poly}(n))$ bits.
- The functions $\{M_i\}$ do not mix between components. That is, for all i and a, if $x \in S_a$ then $M_i(x) \in S_a$ as well.
- The functions $\{M_i\}$ instantly mix within each component. That is, for all a and $x \in S_a$, if $i \in_R \text{Ind}_M$, then the total variation distance between $M_i(x)$ and a uniform sample from S_a is no more than 2^{-n-2}.

The last condition is often easy to satisfy directly. For graph isomorphism, if Ind_M were the set of permutations, then M_i would apply the permutation i to a graph. For group membership, if Ind_M were a set of sequences of coin flips that could generate nearly uniform samples from the subgroup (e.g. the coin flips could encode straight-line programs as in [37]), then M_i would multiply its argument by the element of the subgroup implied by the coin flips. In general, given a Markov chain over S that does not mix between components but mixes rapidly over each component, each step of which consists of choosing uniform random sample from a set of invertible rules and applying that rule, then iterating that Markov chain to amplify its spectral gap will give a component mixer.

In graph isomorphism, generating a random graph, testing whether some encoding of a graph is valid, and generating a random permutation to apply to the

vertices are all easy. Similarly, in group membership, generating a random element of the whole group (as opposed to the subgroup) is easy, as is generating a random element of the subgroup. When we abstract these problems to component mixer problems, we want the corresponding operations to be easy as well. This leads to our definition of query access to a component mixer.

Definition 4. An algorithm has query access to a component mixer $\{M_i\}$ if the algorithm can do each of the following operations in one query with failure probability no more than 2^{-n}.

- Test an n-bit string for membership in S.
- Generate an uniform random sample from S.
- Test a string for membership in Ind_M.
- Generate an uniform random sample from Ind_M.
- Given $s \in S$ and $i \in \text{Ind}_M$, compute $M_i(s)$.
- Given $s \in S$ and $i \in \text{Ind}_M$, compute $M_i^{-1}(s)$.

If we are considering *quantum* algorithms, we want to give the algorithm some quantum power. For example, in graph isomorphism, generating a uniform quantum superposition of *all* graphs is easy, as is generating a uniform quantum superposition of all members of the permutation group [33]. We give quantum component algorithms the equivalent powers.

Definition 5. An algorithm has quantum query access to a component mixer $\{M_i\}$ if the algorithm can do each of the following operations coherently in one query with failure probability no more than 2^{-n}:

- Test an n-bit string for membership in S.
- Generate the state $\sum_{s \in S} |s\rangle$ or measure the projector onto that state.
- Test a string for membership in Ind_M.
- Generate the state $\sum_{i \in \text{Ind}_M} |i\rangle$ or measure the projector onto that state.
- Compute the "controlled-M" operator, abbreviated CM. CM takes three registers as input: the first is the number -1, 0, or $+1$, the second is a string i, and the third is an n-bit string s. On input $|\alpha, i, s\rangle$, $\text{CM}\,|\alpha, i, s\rangle = |\alpha, i, M_i^\alpha(s)\rangle$ if $i \in \text{Ind}_M$ and $s \in S$; otherwise $\text{CM}\,|\alpha, i, s\rangle = |\alpha, i, s\rangle$.

As a technical detail, we assume that any algorithm given (quantum) query access to a component mixer $\{M_i\}$ knows both n and the number of bits needed to encode an element of Ind_M.

We can now state the definitions of our query problems.

Definition 6. The same component problem is: given query access to a component mixer $\{M_i\}$ on a set S and two elements $(s,t) \in S$, accept if s and t are in the same component of S.

Definition 7. The multiple components problem is: given query access to a component mixer $\{M_i\}$ on a partition $\{S_1, \ldots, S_c\}$, accept if $c > 1$.

Definition 8. The multiple balanced components problem is: There is a partition $\{S_1, \ldots, S_c\}$ with the promise that either there is only one component or no component contains more than half the elements in S. Given query access to a component mixer $\{M_i\}$ on that partition and the string 0^n, accept if $c > 1$.

On a quantum computer, we can also try to generate the uniform superposition over a component.

Definition 9. The component superposition problem is: given quantum query access to a component mixer $\{M_i\}$ on a set S and an element $s \in S$, output the state

$$|S_j\rangle = \frac{1}{\sqrt{|S_j|}} \sum_{u \in S_j} |u\rangle$$

where S_j is the (unknown) component containing s.

The decision problems can also be viewed as complexity classes. We define the class SCP to be the set of languages that are polynomial-time reducible to the same component problem with bounded error. Similarly, we define MCP by reference to multiple components and MBCP by reference to multiple balanced components.

5.3. Basic properties of component mixers

Lemma. (Component mixers are fully connected) *If s and t are in the same component, then there exists i such that $t = M_i(s)$.*

Proof. Assume the contrary. Suppose s and t are in the same component S_j and let $A = \{M_i(s) : i \in \mathrm{Ind}_M\}$. By assumption, $t \notin A$. This means that the variation distance between $M_i(s)$ (for $i \in_R \mathrm{Ind}_M$) and a uniform sample on S_j is

$$\begin{aligned}
&\frac{1}{2} \sum_{u \in S_j} \left| \Pr\left[M_i(s) = u\right] - \frac{1}{|S_j|} \right| \\
\geq &\frac{1}{2} \left| \Pr\left[M_i(s) = t\right] - \frac{1}{|S_j|} \right| \\
= &\frac{1}{2\,|S_j|} \\
\geq &2^{-n-1} > 2^{-n-2},
\end{aligned}$$

which contradicts the fact that M is a component mixer. □

A uniform quantum superposition over all the elements in one component is a potentially useful state. It is not obvious whether a quantum computer can produce or verify such a state with a small number of queries to a component mixer, but it is possible to verify that a state is in the span of such superpositions.

LEMMA. (Quantum computers can project onto component superpositions) *A quantum computer can, with a constant number of queries to a component mixer, measure the projector*

$$P = \sum_k \left(|S_k|^{-1/2} \sum_{x \in S_k} |x\rangle \right) \left(|S_k|^{-1/2} \sum_{x \in S_k} \langle x| \right) = \sum_k \left(\frac{1}{|S_k|} \sum_{x,y \in S_k} |x\rangle\langle y| \right)$$

with negligible error as a function of n.

Proof. Starting with a state $|\psi\rangle$, we give an algorithm to measure P on $|\psi\rangle$. The algorithm uses three registers: $|\psi\rangle$ starts in register A; registers B and C are ancillas. B's computational basis is Ind_M and C holds a single bit. To simplify the notation, we write the uniform superposition in register B as $|e_0\rangle = |\mathrm{Ind}_M|^{-1/2} \sum_i |i\rangle_M$. The algorithm is:

1. Initialize register B to $|e_0\rangle_B$ and register C to $|0\rangle$. This gives the state
$$|\varphi_1\rangle = |\psi\rangle_A |e_0\rangle_B |0\rangle_C.$$

2. Apply controlled-M with the control set to 1. This is equivalent to applying M unconditionally. Let $\tilde{M}_j$ be the quantum operator corresponding to the action of M_j on register A. That is, $\langle s'|\tilde{M}_j|s\rangle = \langle s'|M_j(s)\rangle$. With this notation, the action of this step on registers A and B is $U = \left(\sum_j \tilde{M}_j \otimes |j\rangle\langle j| \right)$. The resulting state is
$$\begin{aligned} |\varphi_2\rangle &= U|\psi\rangle_A |e_0\rangle_B |0\rangle_C \\ &= |e_0\rangle_{BB}\langle e_0|U|\psi\rangle_A |e_0\rangle_B |0\rangle_C + (\mathbb{I} - |e_0\rangle_{BB}\langle e_0|)\, U|\psi\rangle_A |e_0\rangle_B |0\rangle_C \end{aligned}$$

3. Apply the unitary operator $|e_0\rangle_{BB}\langle e_0| \otimes X_C + (\mathbb{I} - |e_0\rangle_{BB}\langle e_0|) \otimes I_C$. This sets register C to $|1\rangle$ is register B is still in the state $|e_0\rangle$. The state is now
$$|\varphi_3\rangle = |e_0\rangle_{BB}\langle e_0|U|\psi\rangle_A |e_0\rangle_B |1\rangle_C + (\mathbb{I} - |e_0\rangle_{BB}\langle e_0|)\, U|\psi\rangle_A |e_0\rangle_B |0\rangle_C.$$

4. Uncompute step 2 by applying $U^\dagger$. This gives
$$\begin{aligned} |\varphi_4\rangle =& U^\dagger |e_0\rangle_{BB}\langle e_0|U|\psi\rangle_A |e_0\rangle_B |1\rangle_C \\ &+ U^\dagger (\mathbb{I} - |e_0\rangle_{BB}\langle e_0|)\, U|\psi\rangle_A |e_0\rangle_B |0\rangle_C \\ =& \left(U^\dagger |e_0\rangle_{BB}\langle e_0| U \right) |\psi\rangle_A |e_0\rangle_B |1\rangle_C \\ &+ \left(\mathbb{I} - U^\dagger |e_0\rangle_{BB}\langle e_0| U \right) |\psi\rangle_A |e_0\rangle_B |0\rangle_C. \end{aligned}$$

To simplify this result, observe that the matrix $|\mathrm{Ind}_M|^{-1} \sum_j \tilde{M}_j$ is the Markov matrix obtained by applying one of the M_i uniformly at random to an element of S. From the definition of a component mixer, $|\mathrm{Ind}_M|^{-1} \sum_j \tilde{M}_j \approx P$. Furthermore, $P|\psi\rangle$ has the form $\alpha \sum_{x \in S_a} |x\rangle$ for some a and α, and M_k preserves the set S_a, so $\tilde{M}_k P|\psi\rangle = P|\psi\rangle$ for all k. Using these observations, we can simplify

$$
\begin{aligned}
&U^\dagger |e_0\rangle_{BB}\langle e_0| U |\psi\rangle_A |e_0\rangle_B \\
=&\left(\sum_k \tilde{M}_k^\dagger \otimes |k\rangle_{BB}\langle k|\right) |e_0\rangle_{BB}\langle e_0| \left(\sum_j \tilde{M}_j \otimes |j\rangle\langle j|\right) |\psi\rangle_A |e_0\rangle_B \\
=&\left(\sum_k \tilde{M}_k^\dagger \otimes |k\rangle_{BB}\langle k|\right) |e_0\rangle_B \left|\operatorname*{Ind}_M\right|^{-1} \left(\sum_j \tilde{M}_j\right) |\psi\rangle_A \\
\approx&\left(\sum_k \tilde{M}_k^\dagger \otimes |k\rangle_{BB}\langle k|\right) P|\psi\rangle_A |e_0\rangle_B \\
=&P\left(\sum_k |k\rangle_{BB}\langle k|\right) |\psi\rangle_A |e_0\rangle_B \\
=&P|\psi\rangle_A |e_0\rangle_B .
\end{aligned}
$$

Plugging this in, we have

$$|\varphi_4\rangle \approx P|\psi\rangle_A |e_0\rangle_B |1\rangle_C + (\mathbb{I} - P)\,|\psi\rangle_A |e_0\rangle_B |0\rangle_C$$

with negligible error.

At this point, register B is unentangled with the rest of the system, register C contains the outcome of the measurement, and register A contains the correct final state. □

On a quantum computer, SAME COMPONENT reduces to COMPONENT SUPERPOSITION: given two initial elements, a swap test can decide with bounded error whether their respective component superpositions are the same state or non-overlapping states.

5.4. Placing component mixer problems in the complexity zoo

5.4.1. Inclusions

Several of the complexity class relationships in Table 5.1 are straightforward. MULTIPLE COMPONENTS is a relaxation of MULTIPLE BALANCED COMPONENTS, so MBCP $\subseteq$ MCP. The "component mixers are fully connected" lemma implies a

simple NP algorithm for SAME COMPONENT, so SCP $\subseteq$ NP. MULTIPLE COMPONENTS can be restated as "do there exist two objects that are *not* in the same component?", so MCP $\subseteq$ NP$^{\text{co-SCP}}$ and hence MCP $\subseteq$ NP$^{\text{co-NP}}$.

In the appendix, we give two Arthur-Merlin protocols for MULTIPLE BALANCED COMPONENTS:

- A protocol to prove a "yes" answer. In this protocol, Merlin solves the SAME COMPONENT problem on input given by Arthur (appendix 5.7.1).
- A protocol to prove a "no" answer (appendix 5.7.2).

The existence of these protocols implies that MBCP $\subseteq$ AM, BPP$^{\text{SCP}}$, and co-AM. We also give a QMA protocol for MULTIPLE COMPONENTS (see appendix 5.7.3).

SAME COMPONENT is reducible to STATISTICAL DIFFERENCE: to test whether s and t are in the same component, choose $i, j \in_R \text{Ind}_M$ and test whether $M_i(s)$ and $M_j(t)$ have the same distribution. STATISTICAL DIFFERENCE is complete for SZK, so SCP $\subseteq$ SZK [38].

MULTIPLE BALANCED COMPONENTS is also reducible to STATISTICAL DIFFERENCE: choose $a, b \in_R S$ and $i, j \in_R \text{Ind}_M$. If there are multiple balanced components, then the predicate that the first two and last two elements of $(a, M_i(a), b, M_j(b))$ are in the same component holds w.p. 1, whereas the same predicate holds on four independent uniform samples from S w.p. at most ¼. This means that the variation distance between $(a, M_i(a), b, M_j(b))$ and four independent samples is at least ¾. If, on the other hand, there is only one component, then $(a, M_i(a), b, M_j(b))$ is negligibly different four independent samples from S. Therefore, MULTIPLE BALANCED COMPONENTS reduces to STATISTICAL DIFFERENCE on the distribution of $(a, M_i(a), b, M_j(b))$ versus four independent uniform samples from S. Hence MBCP $\subseteq$ SZK.

5.4.2. Separations

SCP contains group membership (relative to any oracle) and group membership is not in co-MA for black-box groups [35], so SCP $\not\subseteq$ co-MA relative to an oracle.

The quantum query complexity of MULTIPLE COMPONENTS is exponential by reduction from the Grover problem (see appendix 5.8). This implies the existence of an oracle separating MCP and BQP.

5.4.3. Conjectured separations

We conjecture that there is no QCMA or co-QCMA proof for MULTIPLE COMPONENTS or even MULTIPLE BALANCED COMPONENTS This would imply the existence of an oracle separating MBCP from QMA and hence QCMA from QMA.

We further conjecture that MULTIPLE BALANCED COMPONENTS has superpolynomial randomized and quantum query complexity. This conjecture would imply that MBCP is separated from BPP and BQP by an oracle.

5.5. A hardness result for counterfeiting quantum money

We are now ready to prove a hardness result for counterfeiting quantum money. Recall that the quantum money state is defined [3, 2] as

$$|\$_\ell\rangle = \sum_{x \in S_\ell} |x\rangle$$

where S_ℓ is a component of a partition of a big set S and an adversary is given access to a component mixer for that partition. Unlike the other component mixer problems we have discussed, an adversary also has access to a labeling function L that maps each element of S to a label that identifies which component contains that element.

We show that, if an attacker is given one copy of $|\$_\ell\rangle$ and *measures* it in the computational basis, then, under reasonable assumptions, the attacker cannot recreate the state. That is, given some $s \in S_\ell$ (i.e. the measurement outcome), it is hard to produce $|\$_\ell\rangle$. We call this type of attack SIMPLE COUNTERFEITING. Our assumption is that the quantum query complexity of SAME COMPONENT is superpolynomial.

DEFINITION 10. The SIMPLE COUNTERFEITING problem is: given quantum query access to a component mixer $\{M_i\}$ on a set S, quantum query access to a function L that maps each element of S to a unique label identifying the component containing that element, and an element $s \in S$, output the state

$$|S_j\rangle = \frac{1}{\sqrt{|S_j|}} \sum_{u \in S_j} |u\rangle$$

where S_j is the component containing s.

SIMPLE COUNTERFEITING is the same problem as COMPONENT SUPERPOSITION except that the algorithm also has access to the labeling function. This makes the problem seem easier; for example, SAME COMPONENT and MULTIPLE BALANCED COMPONENTS both become trivial with access to the labeling function. We show that the labeling function is unhelpful for the purpose of SIMPLE COUNTERFEITING.

THEOREM. *If the quantum query complexity of COMPONENT SUPERPOSITION is superpolynomial, then the quantum query complexity of SIMPLE COUNTERFEITING is also superpolynomial.*

Proof. The SIMPLE COUNTERFEITING and COMPONENT SUPERPOSITION problems are almost the same: they differ only in that SIMPLE COUNTERFEITING is given access to a label that identifies components. Calculating such a label given only a component mixer is at least as hard as solving SIMPLE COUNTERFEITING in the first place, so we won't be able to provide a valid label. The idea behind the proof is to show that a

correct labeling function is not very helpful for solving SIMPLE COUNTERFEITING, and that, given a component mixer, we can efficiently provide a label that is indistinguishable from a valid label in polynomial time.

We assume for contradiction that we have a quantum query algorithm "alg" that solves SIMPLE COUNTERFEITING in n^k queries for sufficiently large n. Alg is given quantum query access to a component mixer and labeling function and it is promised that the labeling function is consistent with the component mixer. It takes as input an element $s \in S_j$ for some j. It makes n^k quantum queries and produces a mixed state ρ as output. The trace distance between ρ and the desired output state $\frac{1}{\sqrt{|S_j|}} \sum_{u \in S_j} |u\rangle$ is a negligible function of n.

We give an algorithm that solves COMPONENT SUPERPOSITION with high probability using alg as a subroutine.

As input, we have quantum query access to a component mixer on n bits and an n-bit string s. This means that the space of n bit strings is partitioned into components $S_1, \ldots, S_c$ and a set of "garbage" strings $G = \{0,1\}^n \setminus (S_1 \cup \cdots \cup S_c)$, where c is the (unknown) number of components. We are not given access to a labeling function. WLOG, we assume that $s \in S_1$.

We define an instance of SIMPLE COUNTERFEITING on $2n$-bit strings that can be used to solve the original COMPONENT SUPERPOSITION problem. To simplify the notation, we treat each $2n$-bit string as a pair of binary numbers, each between 0 and $2^n - 1$. In our instance of SIMPLE COUNTERFEITING, the components are $\{0\} \times S_1, \ldots, \{0\} \times S_c$ and $\{0\} \times G$. Each other element (that is, everything that has something nonzero as its first n bits) is its own component. We use the component mixer

$$M_i^{(0)}(r,z) = \begin{cases} (0, M_i(z)) & \text{if } r = 0 \\ (r,z) & \text{otherwise} \end{cases}$$

and *incorrect* label

$$L^{(0)}(r,z) = \begin{cases} (0,0) & \text{if } r = 0 \\ (r,z) & \text{otherwise} \end{cases}.$$

The label $L^{(0)}$ violates the promise of SIMPLE COUNTERFEITING (it assigns the same label to all the components in the original component mixer), so the SIMPLE COUNTERFEITING algorithm run directly on $\{M_i^{(0)}\}$ and $L^{(0)}$ might fail. However, the only way to detect that $L^{(0)}$ is invalid is to query it on some input of the form $(0,t)$ for $t \in S_2 \cup \cdots \cup S_c$. Those inputs are an exponentially small fraction of the domain of $L^{(0)}$ and we can hide them by randomly permuting $L^{(0)}$ and $M_i^{(0)}$, giving this algorithm: □

1. Choose independent random permutations π and σ on $\mathbb{Z}_{2^n} \times \mathbb{Z}_{2^n}$. π indicates where each $2n$-bit string is hidden in the permuted problem and σ scrambles

the labels. (These permutations will take an exponential number of bits to specify, but they can be implemented with no queries to $\{M_i\}$.)

a) Run alg on $\left\{\pi \circ M_i^{(0)} \circ \pi^{-1}\right\}$ and $\sigma \circ L^{(0)} \circ \pi^{-1}$ with the initial element $\pi(0, s)$.

b) Apply π^{-1} coherently to the quantum state that alg produces.

c) Output the last n qubits of the result.

Proof. If $\sigma \circ L^{(0)} \circ \pi$ were a valid label function for the component mixer

$$\left\{\pi \circ M_i^{(0)} \circ \pi\right\},$$

then this algorithm would succeed on each try w.p. negligibly different from 1. We will prove that the invalidity of the labeling function is well enough hidden that the algorithm works anyway.

To prove this, we assume the contrary: there is some $\{M_i\}$ for which this algorithm fails with non-negligible probability. This means that the actual output of our algorithm differs non-negligibly in trace distance from the desired output. Such a difference would be detectable if we knew what the correct output was; we will show that this is impossible by solving the Grover problem more quickly than is allowed by the BBBV theorem using alg as a subroutine.

We generalize the functions $M_i^{(0)}$ and $L^{(0)}$ to a larger family that encodes a Grover search problem. We can picture $\{M_i^{(0)}\}$ as an embedding of the original problem in the first row of a grid in which the first n bits is the row index and the last n bits is the column index (see Figure 5.1 – the unmarked squares are their own components). There are many other ways we could have embedded the original problem, though. (These other embeddings are well-defined, but they are difficult to calculate without access to a labeling function for the original problem.) In particular, we could have placed everything except S_1 on a different row. If we put the other components on the j^{th} row, we get

$$L^{(j)}(r, z) = \begin{cases} (0,0) & \text{if } r = 0 \text{ and } z \in S_1 \\ (0,0) & \text{if } r = j \text{ and } z \notin S_1 \\ (r, z) & \text{otherwise} \end{cases}$$

and

$$M_i^{(j)}(r, z) = \begin{cases} (0, M_i(z)) & \text{if } r = 0 \text{ and } z \in S_1 \\ (j, M_i(z)) & \text{if } r = j \text{ and } z \notin S_1 \\ (r, z) & \text{otherwise} \end{cases}.$$

Alternatively, we could leave them out entirely, giving

$$L^{\text{nowhere}}(r,z) = \begin{cases} (0,0) & \text{if } r = 0 \text{ and } z \in S_1 \\ (r,z) & \text{otherwise} \end{cases}$$

and

$$M_i^{\text{nowhere}}(r,z) = \begin{cases} (0, M_i(z)) & \text{if } r = 0 \text{ and } z \in S_1 \\ (r,z) & \text{otherwise} \end{cases}.$$

We can't efficiently implement queries to L^{nowhere}, M^{nowhere}, $L^{(j)}$ or $M_i^{(j)}$ for $j \neq 0$, but, if we could and if alg didn't notice that the label function was invalid, then the output on any of instances with starting element $(0, s)$ would be

$$\sum_{z \in S_1} |0\rangle|z\rangle.$$

The latter n qubits this state is exactly what we want.

The function L^{nowhere} is a valid labeling function, but all of the $L^{(j)}$ are invalid because they take the same value on the images of $S_1, \ldots, S_c$ even though they are in different components. Nonetheless, they look valid as long as no one ever queries them on the images of $S_2, \ldots, S_c$, which collectively represent less than a 2^{-n} fraction of all possible queries.

We formalize this notion by a reduction from the Grover problem. Suppose $g : \mathbb{Z}_{2^n} \to \{0,1\}$ is a function that outputs 1 at most one input. By the BBBV theorem [34], the query complexity of distinguishing a random point function g from all zeros is $O\left(2^{n/2}\right)$. Using our algorithm for SIMPLE COUNTERFEITING as a subroutine, we will attempt to decide whether g maps any value to 1. We do this by allowing g to select which embedding to use. This gives the "labeling" function

$$L^{[g]}(r,z) = \begin{cases} (0,0) & \text{if } r = 0 \text{ and } z \in S_1 \\ (0,0) & \text{if } g(r) = 1 \text{ and } z \notin S_1 \\ (r,z) & \text{otherwise} \end{cases}$$

and component mixer

$$M_i^{[g]}((r,z)) = \begin{cases} (0, M_i(z)) & \text{if } r = 0 \text{ and } z \in S_1 \\ (j, M_i(z)) & \text{if } g(r) = 1 \text{ and } z \notin S_1 \\ (r,z) & \text{otherwise} \end{cases}.$$

If $g(j) = 1$ for some j, then $L^{[g]} = L^{(j)}$ and $M_i^{[g]} = M_i^{(j)}$; otherwise $L^{[g]} = L^{\text{nowhere}}$ and $M^{[g]} = M^{\text{nowhere}}$. It is possible to evaluate either $L^{[g]}(r,z)$ or $M_i^{[g]}(r,z)$ with a very large number of queries to the original component mixer $\{M_i\}$ and *one* query to $g(r)$. (Evaluating the functions coherently requires a second query to $g(r)$ to uncompute garbage.)

If we choose independent random permutations π and σ on $\mathbb{Z}_{2^n} \times \mathbb{Z}_{2^n}$ and run alg on

$$\left\{\pi \circ M_i^{[g]} \circ \pi^{-1}\right\} \text{ and } \sigma \circ L^{[g]} \circ \pi - 1$$

with initial state $\pi(0, s)$, the output of the algorithm is some mixed state that depends on g. Let ρ_0 be the density matrix of that mixed state if g is all zeros and let ρ_{point} be the density matrix if g is a uniformly random point function. □

Claim. $\left\|\rho_0 - \rho_{\text{point}}\right\|_{\text{tr}}$ is a negligible function of n.

Proof. Assume the contrary: $\left\|\rho_0 - \rho_{\text{point}}\right\|_{\text{tr}} \geq n^{-k}$ for some fixed k and an infinite sequence of values of n. If we run alg on $L^{[g]}$ and $M_i^{[g]}$, we can then decide whether the output is ρ_0 or ρ_{point} and therefore whether g is all zeros or a point function by measuring the output. We will get the right answer w.p. at least $\frac{1}{2} + \frac{n^{-k}}{2}$. We can amplify n^{2k} times to get the right answer w.p. at least $2/3$ by a Chernoff bound. By assumption, alg makes n^r queries to $L^{[g]}$ and $M_i^{[g]}$. That means that, in $n^{r+2k} = o\left(n^{n/2}\right)$ queries, we can determine whether $g(j) = 1$ for any j; this is impossible by the BBBV theorem. Therefore $\left\|\rho_0 - \rho_{\text{point}}\right\|_{\text{tr}}$ is a negligible function of n.

It follows that, if we apply π^{-1} to ρ_0 and to ρ_{point}, the results differ negligibly in trace distance. The result of applying π^{-1} to ρ_0 is the uniform superposition over $\{0\} \times S_1$ up to negligible error because if $g = 0$ then alg's promise is satisfied and it produces the correct answer. Furthermore, if we set $g(0) = 1$, then the output distribution is still ρ_0 because the distribution of component mixers and labels seen by alg is independent of which point function we choose. This means that π^{-1} applied to the output of alg on $\left\{\pi \circ M_i^{(0)} \circ \pi^{-1}\right\}$ and $\sigma \circ L^{(0)} \circ \pi^{-1}$ with initial state $\pi(0, s)$ differs negligibly from the uniform superposition over $\{0\} \times S_1$ in trace distance.

This contradicts the assumption that there exists some input on which our algorithm fails, so our algorithm solves COMPONENT SUPERPOSITION with negligible error. □

We can replace the assumption that COMPONENT SUPERPOSITION is hard with the assumption that SAME COMPONENT is hard because SAME COMPONENT reduces to COMPONENT SUPERPOSITION.

5.6. Open problems

There are a number of open problems related to this work.

Ideally, we would prove the impossibility of more general forms of counterfeiting. If we could show that, given one copy of $|\$_\ell\rangle$ for some ℓ, it is hard to produce a second copy of $|\$_\ell\rangle$, then we would know that (in a black-box model) quantum

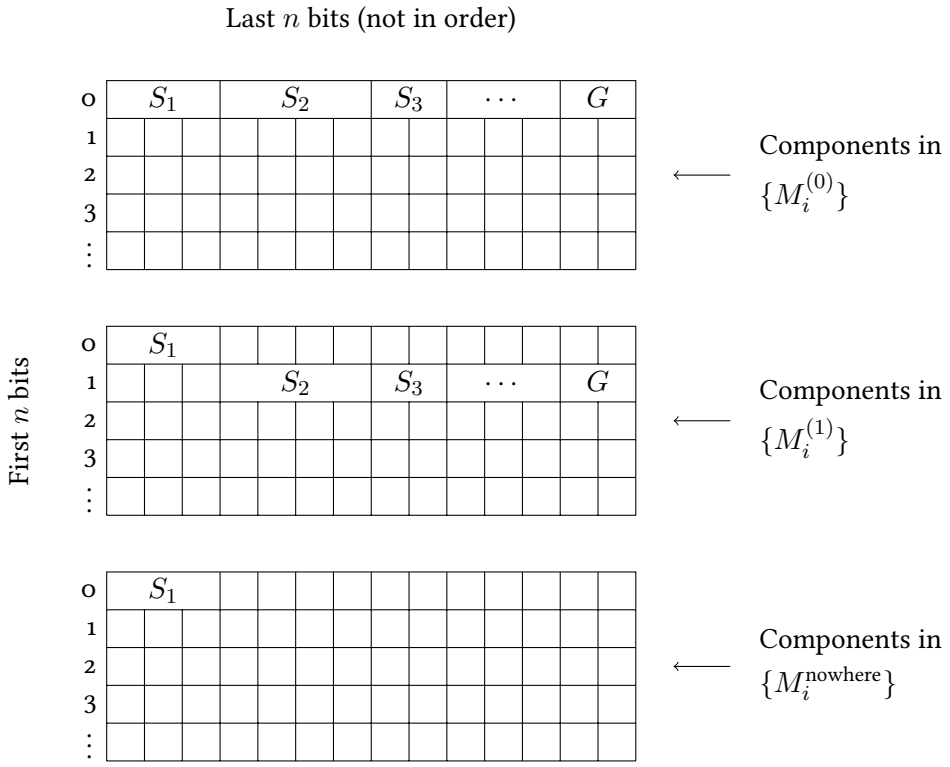


Figure 5.1: There are $2^n + 1$ ways to hide components that are hard to label.

money could not be counterfeited. An even better result would be collision-freedom: that is hard for anyone to produce a state of the form $|\$_\ell\rangle \otimes |\$_\ell\rangle$ by any means, even for a random ℓ of an attacker's choice. (Collision-freedom implies that copying is impossible: if an attacker could copy a given quantum money state, then the output of the algorithm would be contain two copies of $|\$_\ell\rangle$ for the value of ℓ implied by the input.)

It should be possible to prove quantum lower bounds on the query complexity of SAME COMPONENT and MULTIPLE BALANCED COMPONENTS. This would strengthen the hardness result for counterfeiting quantum money.

A classical oracle separating MCP and QCMA would also separate QMA and QCMA. We conjecture that an appropriate worst-case component mixer would work, but we have no proof.

A cryptographically secure component mixer could be a useful object, and a good cryptographically secure component mixer with an associated labeling function would give a better quantum money protocol than quantum money from knots. (Knot invariants have all kinds of unnecessary properties.) If we had that as well as a hardness result for generating quantum money collisions, then quantum money would be on a sound theoretical footing.

5.7. Query protocols for component problems

5.7.1. An AM query protocol for MULTIPLE BALANCED COMPONENTS

Suppose that Merlin wants to prove to Arthur that some component mixer has multiple balanced components. Arthur and Merlin run this protocol:

	Arthur	Merlin
1.	Choose $s_1, s_2 \in_R S$, $i \in_R \{1,2\}$ and $j \in_R \mathrm{Ind}_M$.	
2.	Compute $t = M_j(s_i)$.	
3.	Send s_1, s_2, t to Merlin.	
4.		If s_1 and s_2 are in different components, compute $i' = i$. Otherwise, choose $i' \in_R \{1,2\}$.
5.		Send i' to Arthur.
6.	Accept iff $i = i'$.	

If $\{M_i\}$ has multiple balanced components, then with probability at least ½, s_1 and s_2 are in different components. In this case, Merlin will always answer correctly. This means that Merlin is correct w.p. at least ¾. If, on the other hand, M has only one component, then t is a nearly uniform sample from S (trace distance at most $2^{-n-2} \leq \frac{1}{8}$). This means that Merlin can guess i correctly with probability at

most $\frac{5}{8}$. With constant overhead, this protocol can be amplified to give soundness and completeness errors $\frac{1}{3}$.

Steps 1, 2, 3, 5, and 6 can be done in a constant number of queries to the component mixer oracle. Step 4 requires Merlin to solve the SAME COMPONENT to decide whether t is in the same component as s_1, s_2, or both. This means that if Arthur had the power of SCP (with oracle access to $\{M_i\}$), then he could run the protocol on his own.

5.7.2. A co-AM query protocol for MULTIPLE BALANCED COMPONENTS

Suppose that Merlin wants to prove to Arthur that some component mixer has a single component (as opposed to multiple balanced components). Arthur and Merlin run this protocol:

	Arthur	Merlin
1.	Choose $s_1, s_2 \in_R S$.	
2.	Send s_1, s_2 to Merlin.	
3.		Choose $i \in_R \mathrm{Ind}_M$ such that $M_i(s_1) = s_2$.
4.		Send i to Arthur.
5.	Accept iff $M_i(s_1) = s_2$.	

If there is only one component, then s_1 and s_2 are in the same component and Merlin can find i because component mixers are fully connected. If, on the other hand, there are multiple balanced components, then w.p. at least $\frac{1}{2}$, s_1 and s_2 are in different components and no such i exists.

This means that this proof is complete and has soundness error at most $\frac{1}{2}$. A constant amount of amplification will reduce the soundness error below $\frac{1}{3}$.

5.7.3. A quantum witness for MULTIPLE COMPONENTS

Given a "yes" instance of MULTIPLE COMPONENTS problem, let S_1 and S_2 be two distinct components. Then a valid witness state is

$$|\psi_{\mathrm{MC}}\rangle = \left(\sum_{s \in S_1} |s\rangle\right) \otimes \left(\sum_{s \in S_2} |s\rangle\right).$$

To verify the witness, Arthur first measures the projector of each register onto the space of uniform superpositions over components (see section 5.3). If either measurement outputs zero, Arthur rejects. Otherwise Arthur performs a swap test between the two registers and accepts iff the swap test says that the registers are different.

On a valid witness, Arthur's projections succeed with probability close to 1. The states in the two registers have disjoint support (both before and after the swap test), so the swap test indicates that the states are different w.p. ½. Arthur therefore accepts a valid witness w.p. ½.

If there is only one component then projecting onto the space of uniform superpositions over components is equivalent to projecting onto the uniform superposition over S. Therefore, on any witness, if Arthur's projections succeed then the post-measurement state is (up to negligible error) two copies of the uniform superposition over S. Those two copies are approximately the same state, so the swap test says that they are the same and Arthur rejects w.p. near 1. Standard techniques can amplify this protocol to give completeness and soundness errors less than ⅓.

5.8. Multiple components has exponential quantum query complexity

We can embed an instance of the Grover problem into MULTIPLE COMPONENTS. Let g be the instance of the Grover problem on n bits (i.e. $g : \mathbb{Z}_{2^n} \to \{0,1\}$ is either all zeros or a point function). Let $\mathrm{Ind}_M = \mathbb{Z}_{2^n}$ and define the component mixer

$$M_i(x) = \begin{cases} (x+i) \bmod 2^n & \text{if } g(x) = g(x+i) = 0 \\ x & \text{otherwise} \end{cases}.$$

If g is all zeros then there is a single component, but if $g(y) = 1$ then y is in its own component. The function M_i can be evaluated with two queries to g, so the Grover decision problem on g reduces to MULTIPLE COMPONENTS on $\{M_i\}$.

Hence, by the BBBV theorem [34], the quantum query complexity of MULTIPLE COMPONENTS is $\Omega\left(2^{n/2}\right)$.

6. Quantum money from knots

In this chapter, we present quantum money based on knots (see Figure 6.1). Quantum money from knots is a variant of the component mixer quantum money discussed in the previous chapter. To a first approximation, the large set S is the set of knot diagrams. This set is partitioned into equivalence classes of knots diagrams, and the labeling function L is a knot invariant called the Alexander polynomial [39]. The component mixers are sequences of Reidemeister moves that convert knot diagrams into other equivalent diagrams. The purported security of our quantum money scheme is based on the assumption that given two different looking but equivalent knots, it is difficult to explicitly find a transformation that takes one to the other.

Figure 6.1: Quantum money from knots

This description is only a rough approximation. Knot diagrams are continuous objects that are tedious to work with on a computer, so we use a different representation called planar grid diagrams instead. For technical reasons, we use oriented links instead of knots. An oriented link is one or more knots, possible interlinked, where each knot has an associated orientation. The set of oriented links and the set of planar grid diagrams are infinite, so we cut off the set at some size. That size is our security parameter. That cutoff introduces nonuniform weights into our superposition and component mixer, so we need to tweak the verification algorithm using ideas from the Metropolis sampling algorithm [40]. Finally, the Alexander polynomial is not a consistent labeling function: many equivalence classes of oriented links will have the same Alexander polynomial. We conjecture that this flaw does not weaken the security of our scheme.

6.1. Knots, links, and grid diagrams

In this section we briefly review the standard concepts of knots and links and how they are represented using diagrams. The same knot (or link) can be represented

as a diagram in many different ways; the Reidemeister moves are a set of local changes to a diagram that do not change the underlying knot or link. We will review how to compute the Alexander polynomial for a given oriented link diagram in polynomial time. The Alexander polynomial is a link invariant in the sense that if we compute its value on diagrams that represent the same oriented link we get the same polynomial. Finally, we review planar grid diagrams and grid moves, which implement Reidemeister moves on grid diagrams.

6.1.1. Knots and links

We can think of a knot as a loop of string in 3 dimensions, that is, a map from S^1 into $\mathbb{R}^3$. Since it is hard to draw knots in 3 dimensions, usually a knot is represented by a projection of the 3 dimensional object into 2 dimensions where, at each crossing, it is indicated which strand passes above and which below. This is called a knot diagram. In this paper we will be interested in links, which correspond to one or more loops of string (called the components of the link). An oriented link is a link that has a direction associated with each component. An example of an oriented link diagram is shown in figure 6.2.

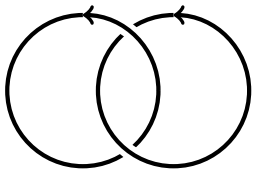


Figure 6.2: An oriented link diagram.

Two links (or knots) are equivalent if one can be smoothly morphed into the other without cutting the string. If unoriented links K_1 and K_2 are equivalent and they are represented by diagrams D_1 and D_2 respectively, then diagram D_1 can be transformed into D_2 (and vice versa) using the Reidemeister moves pictured in figure 6.3. (For oriented links the Reidemeister moves can be drawn with the same pictures as in figure 6.3 but with orientations along the edges that are consistent before and after applying the move). Looking at these moves, one sees that if two diagrams can be brought into the same form by using the moves then the diagrams represent equivalent links. The converse of this statement is a theorem.

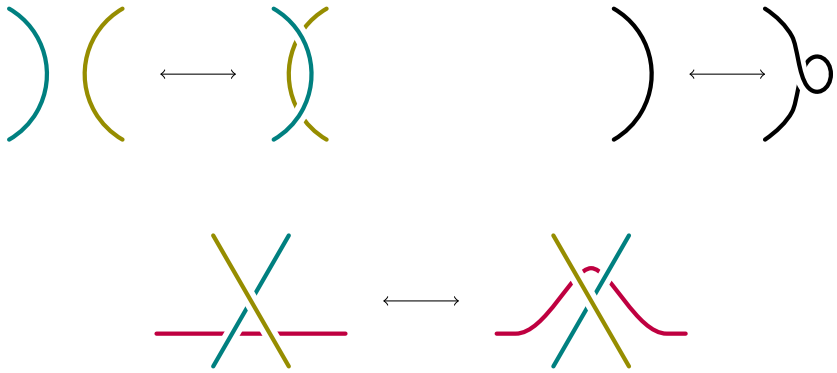


Figure 6.3: The three Reidemeister moves for unoriented link diagrams.

6.1.2. The Alexander polynomial of an oriented link

The Alexander polynomial is a polynomial $\Delta(x)$ that can be computed from a given oriented link diagram and is invariant under the Reidemeister moves. In this section, following Alexander [39], we describe how to compute this polynomial. It will be clear from our discussion that the computation of $\Delta(x)$ can be done in polynomial time in the number of crossings of the diagram.

Suppose we are given a diagram of an oriented link L. If the diagram is disconnected, apply the first Reidemeister move in Figure 6.3 to connect the diagram. Let us write a for the number of crosses in the diagram. The curve of the diagram then divides the two dimensional plane into $a+2$ regions including one infinite region (this follows from Euler's formula). The following recipe can be used to calculate $\Delta(x)$:

1. For each region $i \in \{1, \ldots, a+2\}$, associate a variable r_i.
2. For each of the a crossings, write down an equation
$$xr_j - xr_k + r_l - r_m = 0,$$
where $\{r_j, r_k, r_l, r_m\}$ are the variables associated with the regions adjacent to the crossing, in the order pictured in figure 6.4.
3. Write the set of equations as a matrix equation
$$\mathcal{M} \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_{m+2} \end{pmatrix} = 0.$$

This defines the matrix $\mathcal{M}$ which has a rows and $a+2$ columns. The entries of $\mathcal{M}$ are elements of the set

$$\{\pm 1, \pm x, 1+x, 1-x, -1+x, -1-x\}.$$

4. Delete two columns of $\mathcal{M}$ which correspond to adjacent regions in the link diagram. This gives an $a \times a$ matrix $\mathcal{M}_0$.

5. Take the determinant of $\mathcal{M}_0$. This is a polynomial in x. Divide this polynomial by the factor $\pm x^q$ chosen to make the lowest degree term a positive constant. The resulting polynomial is the Alexander polynomial $\Delta(x)$ for the given link.

When we use the Alexander polynomial in this paper, we are referring to the list of coefficients, not the value of the polynomial evaluated at some x.

Figure 6.4: An equation $xr_j - xr_k + r_l - r_m = 0$ is associated with each crossing, where the adjacent regions $\{r_j, r_k, r_l, r_m\}$ are labeled as shown. Note that the labeling of the adjacent regions depends on which line crosses on top.

6.1.3. Grid diagrams

It is convenient to represent knots as *planar grid diagrams*. A planar grid diagram is a $d \times d$ grid on which we draw d X's and d O's. There must be exactly one X and one O in each row and in each column, and there may never be an X and an O in the same cell. We draw a horizontal line in each row between the O and the X in that row and a vertical line in each column between the X and the O in that column. Where horizontal lines and vertical lines intersect, the vertical line always crosses above the horizontal line.

Knots (or links) in a grid diagram carry an implicit orientation: each vertical edge goes from an X to an O, and each horizontal edge goes from an O to an X. Figure 6.5 shows an example of a $d = 4$ planar grid diagram.

A planar grid diagram G can be specified by two disjoint permutations π_X, $\pi_O \in S_d$, in which case the X's have coordinates $\{(i, \pi_X(i))\}$ and the O's have coordinates $\{(i, \pi_O(i))\}$ for $i \in \{1, ..., d\}$. Two permutations are said to be disjoint if for all i, $\pi_X(i) \neq \pi_O(i)$. Any two disjoint permutations $\pi_X, \pi_O \in S_d$ thus define a planar grid diagram $G = (\pi_X, \pi_O)$. Every link can be represented by many different grid diagrams.

We can define three types of grid moves. The grid moves are transformations on planar grid diagrams that, like the Reidemeister moves for link diagrams, are sufficient to generate all planar grid diagrams of the same oriented link.

- The first type of move is a cyclic permutation of either the rows or the columns. Figure 6.6 shows an example of this type of move on columns. We can think of these moves as grabbing both markers in the rightmost column, pulling them behind the page, and putting them back down on the left. These moves are always legal. There are equivalent moves on rows.
- The second type of move is transposition of two adjacent rows or columns. This can be done only when no strand would be broken. In Figure 6.7 we show examples of when this move is allowed. The legality of this move depends only on the position of the markers in the rows or columns being transposed.[1]
- The third type of move adds one row and one column (this is called stabilization) or deletes one row and one column (this is called destabilization), as shown in Figure 6.8. Destabilization selects three markers forming an "L" shape with sides of length 1 and collapses them into one. That is, it deletes one row and one column such that all three markers are removed. The inverse move selects any marker, adds a row and a column adjacent to that marker, and replaces that marker with three new markers. Any X or O can always be legally stabilized and any three markers forming an "L" shape with sides of length 1 can be destabilized unless they form a box (i.e. a 2×2 square with a marker in all four positions).

In the remainder of this paper we will represent links exclusively with planar grid diagrams.

[1] For the case of columns i and $i+1$, the precise condition is that either the two intervals $[\min(x_i, o_i), \max(x_i, o_i)]$ and $[\min(x_{i+1}, o_{i+1}), \max(x_{i+1}, o_{i+1})]$ do not overlap or one of the intervals contains the other.

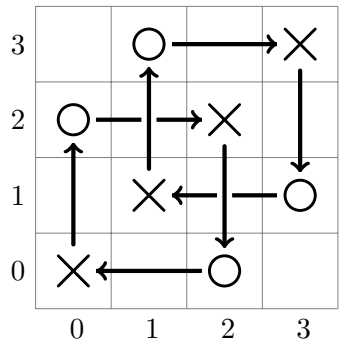


Figure 6.5: A planar grid diagram encodes an oriented link by connecting X and O marks.

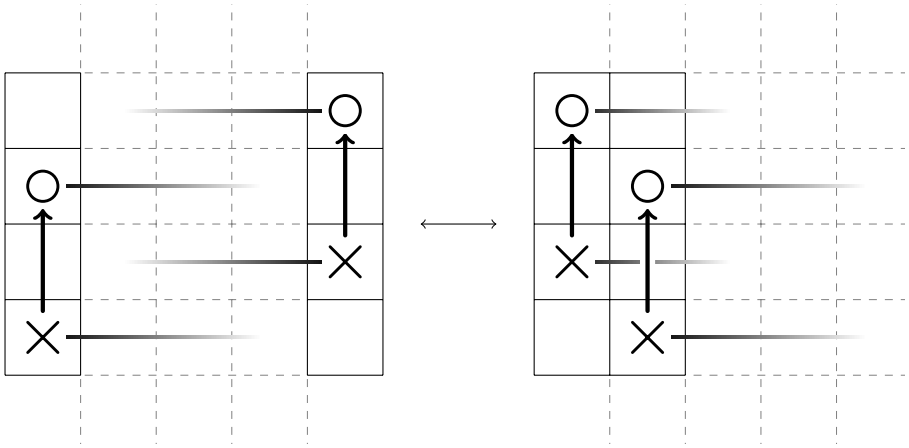


Figure 6.6: Columns can be cyclically permuted. This move is always legal.

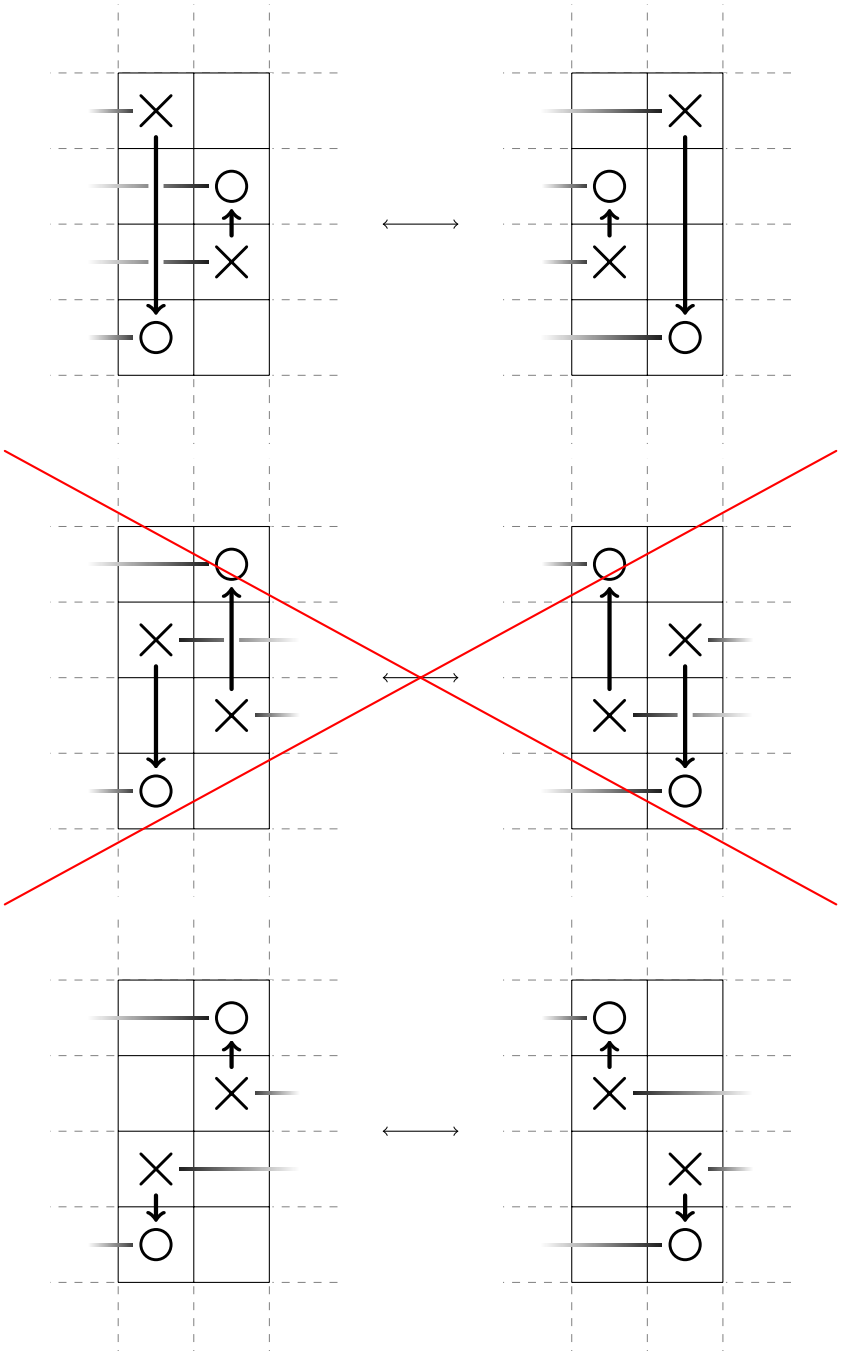


Figure 6.7: Adjacent columns can be transposed. The legality of these moves depends only on the positions of markers in the columns being transposed. The positions of markers in other columns are irrelevant. The middle move is not allowed.

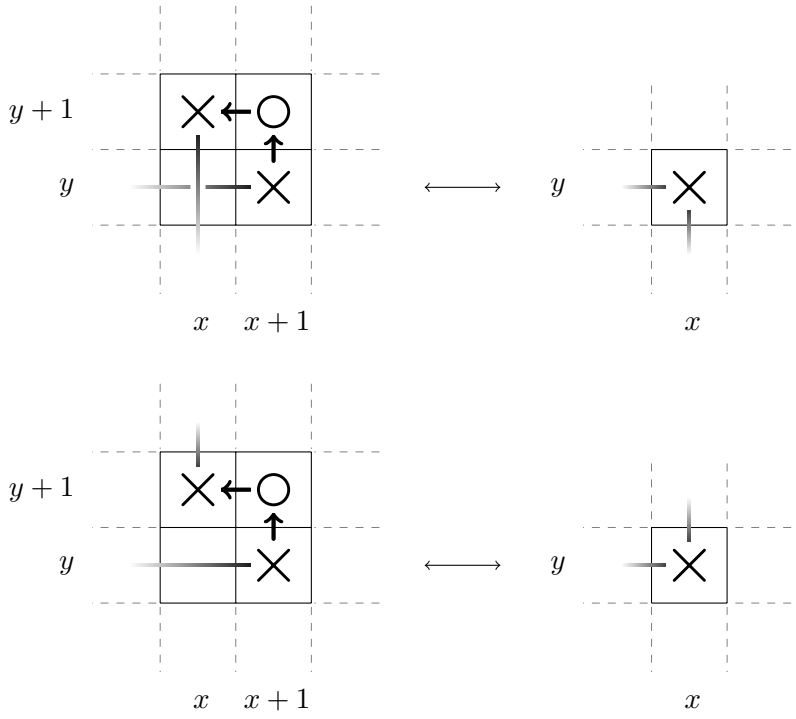


Figure 6.8: Stabilization (right to left) and destabilization (left to right) changes the dimension of the grid. This move is legal when rotated as well as when the markers are switched between X and O. The position of other markers does not matter.

6.2. Quantum money

In this section we describe our quantum money scheme in full detail. In section 6.2.1 we describe how the mint can make quantum money efficiently. In section 6.2.2 we describe the verification procedure, which can be used by anyone (that is, anyone who owns a quantum computer) to ensure that a given quantum bill is legitimate. In section 6.2.3 we discuss the security of the verification procedure. In section 6.2.4 we contrast our quantum money scheme based on knots with a similar but insecure scheme based on graphs.

6.2.1. Minting quantum money

The basis vectors of our Hilbert space are grid diagrams $|G\rangle = |\pi_X, \pi_O\rangle$, where each grid diagram G is encoded by the disjoint permutations π_X, π_O. The di-

mension $d(G)$ is the number of elements in the permutations.[2] The number of disjoint pairs of permutations on d elements is equal to $d!$ times the number of permutations that have no fixed points. The latter quantity is called the number of derangements on d elements, and it is given by $\left[\frac{d!}{e}\right]$ where the brackets indicate the nearest integer function.

To generate quantum money, the mint first chooses a security parameter $\bar{D}$ and defines an unnormalized distribution

$$y(d) = \begin{cases} \frac{1}{d!\left[\frac{d!}{e}\right]} \exp \frac{-(d-\bar{D})^2}{2\bar{D}} & \text{if } 2 \leq d \leq 2\bar{D} \\ 0 & \text{otherwise.} \end{cases}$$

Define an integer valued function

$$q(d) = \left\lceil \frac{y(d)}{y_{min}} \right\rceil,$$

where $\lceil \cdot \rceil$ means round up and where y_{min} is the minimum value of $y(d)$ for $d \in 2, ..., 2\bar{D}$ (we need an integer valued distribution in what follows). The mint then uses the algorithm in [33] to prepare the state (up to normalization)

$$\sum_{d=2}^{2\bar{D}} d! \sqrt{q(d)} |d\rangle.$$

Using a straightforward unitary transformation acting on this state and an additional register, the mint produces

$$\sum_{d=2}^{2\bar{D}} d! \sqrt{q(d)} |d\rangle \left(\frac{1}{d!} \sum_{\pi_X, \pi_O \in S_d} |\pi_X, \pi_O\rangle \right),$$

and then measures whether or not the two permutations π_X and π_O are disjoint. (They are disjoint with probability very close to $1/e$.) If the mint obtains the measurement outcome "disjoint", it uncomputes the dimension d in the first register to obtain the state $|\text{initial}\rangle$ on the last two registers, where

$$|\text{initial}\rangle = \frac{1}{\sqrt{N}} \sum_{\text{grid diagrams } G} \sqrt{q(d(G))} |G\rangle \tag{6.1}$$

and N is a normalizing constant. If the measurement says "not distinct", the mint starts over.

[2] In practice, we will encode d, π_X and π_O as bit strings. Any such encoding will also contain extraneous bit strings that do not describe diagrams; the verification algorithm will ensure that these do not occur.

The distribution $q\,(d)$ is chosen so that if one were to measure $d(G)$ on $|\text{initial}\rangle$, then the distribution of results would be extremely close to Gaussian with mean $\bar{D}$ restricted to integers in the interval $\left[2, 2\bar{D}\right]$. As $\bar{D}$ becomes large, the missing weight in the tails falls rapidly to zero.

From the state $|\text{initial}\rangle$, the mint computes the Alexander polynomial $A(G)$ into another register and then measures this register, obtaining the polynomial p. The resulting state is $|\$_p\rangle$, the weighted superposition of all grid diagrams (up to size $2\bar{D}$) with Alexander polynomial p:

$$|\$_p\rangle = \frac{1}{\sqrt{N}} \sum_{G:A(G)=p} \sqrt{q(d(G))}|G\rangle, \tag{6.2}$$

where N takes care of the normalization. The quantum money consists of the state $|\$_p\rangle$, and the serial number is the polynomial p, represented by its list of coefficients. If the polynomial p is zero, the mint should reject that state and start over. Splittable links have Alexander polynomial equal to zero, and we wish to avoid this case.

6.2.2. Verifying quantum money

Suppose you are a merchant. Someone hands you a quantum state $|\varphi\rangle$ and a serial number which corresponds to a polynomial p and claims that this is good quantum money. To check that indeed this is the case, you would use the following algorithm:

0. Verify that $|\varphi\rangle$ is a superposition of basis vectors that validly encode grid diagrams. If this is the case then move on to step 1, otherwise reject.

1. Measure the Alexander polynomial on the state $|\varphi\rangle$. If this is measured to be p then continue on to step 2. Otherwise, reject.

2. Measure the projector onto grid diagrams with dimensions in the range $[\frac{\bar{D}}{2}, \frac{3\bar{D}}{2}]$. If you obtain $+1$ then continue on to step 3. Otherwise, reject. For valid money, you will obtain the outcome $+1$ with high probability and cause little damage to the state. (We discuss the rationale for this step in section 6.2.3.)

3. Apply the Markov chain verification algorithm described in section 6.2.2.2. If the state passes this step, accept the state. Otherwise, reject.

This procedure is analogous to the general technique for verifying collision-free quantum money based on component mixers. The Markov chain technique we use is somewhat more complicated to the method in the "quantum computers can project onto component superpositions" lemma in Section 5.3: it must account for the fact that we use a *weighted* superposition of basis states.

If the money passes steps 0 and 1 then $|\varphi\rangle$ is a superposition of grid diagrams with the correct Alexander polynomial p. Now passing steps 2 and 3 will (approximately) confirm that $|\varphi\rangle$ is the *correct* superposition of grid diagrams. This procedure will accept genuine quantum money states with high probability, but, as we discuss below, there will be other states that also pass verification. We believe that these states are hard to manufacture.

We now discuss the quantum verification scheme used in step 3. We begin by defining a classical Markov chain.

6.2.2.1. A classical Markov chain

The Markov chain is chosen to have uniform limiting distribution over pairs (G, i) where G is equivalent to (and reachable without exceeding grid dimension $2\bar{D}$ from) the starting configuration, and where $i \in \{1, ..., q(d(G))\}$. Therefore, in the limiting distribution (starting from a grid diagram $\tilde{G}$) the probability of finding a grid diagram G (which is equivalent to $\tilde{G}$) is proportional to $q(d(G))$. We use the extra label i in our implementation of the verifier.

There are two types of update rules for the Markov chain. The first type is an update that changes i while leaving G unchanged (the new value of i is chosen uniformly). The second type is an update that changes G to a new diagram G' while leaving i alone (this type of update is only allowed if $i \leq q(d(G'))$). For the moves that leave i unchanged, our Markov chain selects a grid move at random and proposes to update the current grid diagram by applying that move. The move $G \to G'$ is accepted if the value $i \leq q(d(G'))$.

Recall (from section 6.1.3) that there is a set of grid moves on planar grid diagrams that can be used to transition from one grid diagram to another. These moves only allow transitions between grid diagrams representing equivalent links, and some of these moves change the grid diagram's dimension. In general, any two grid diagrams representing equivalent links can be connected by a sequence of these moves (although finding this sequence is not easy). However, this sequence of moves may also pass through grid diagrams with dimension greater than $2\bar{D}$. In this case, the two equivalent diagrams will not mix, due to the cutoff we defined.

In the appendix we define this Markov chain in detail. This Markov chain does not mix between nonequivalent grid diagrams. As such, it has many stationary distributions. In the next section, we use this Markov chain to produce a quantum verification procedure.

6.2.2.2. The quantum verifier

Let B denote the Markov matrix. As shown in the appendix, B is a doubly stochastic (the sum of the elements in each row and each column is 1) matrix that can be

written as

$$B = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} P_s, \tag{6.3}$$

where each P_s is a permutation on the configuration space of pairs (G, i) such that $i \leq q(d(G))$, and $\mathcal{S}$ is a set of moves that the Markov chain can make at each iteration. We can view the same matrices as quantum operators acting on states in a Hilbert space where basis vectors are of the form $|G\rangle|i\rangle$. For clarity we will refer to the quantum operators as $\hat{B}$ and $\hat{P}_s$.

Our quantum money states $|\$_p\rangle$ live in a Hilbert space where basis vectors are grid diagrams $|G\rangle$. To enable our verification procedure, we define a related state $|\$'_p\rangle$ on two registers, where the second register holds integers $i \in \{0, ..., q_{max}\}$ (here q_{max} is the maximum that $q(d)$ can reach). $|\$'_p\rangle$ is unitarily related to $|\$_p\rangle$, so verifying the former is equivalent to verifying the latter. Define a unitary U that acts on basis vectors in the expanded Hilbert space as

$$U\left(|G\rangle|0\rangle\right) = |G\rangle \frac{1}{\sqrt{q(d(G))}} \sum_{i=1}^{q(d(G))} |i\rangle.$$

To obtain $|\$'_p\rangle$, take the quantum money state $|\$_p\rangle$ and adjoin the ancilla register initialized in the state $|0\rangle$. Then apply U to both registers to produce the state

$$\begin{aligned} |\$'_p\rangle &= U\left(|\$_p\rangle|0\rangle\right) \\ &= \frac{1}{\sqrt{N}} \sum_{G : A(G) = p} \sum_{i=1}^{q(d(G))} |G\rangle|i\rangle. \end{aligned}$$

Note that whereas $|\$_p\rangle$ is a weighted superposition in the original Hilbert space (see equation 6.2), $|\$'_p\rangle$ is a uniform superposition in the expanded Hilbert space. However both $|\$_p\rangle$ and $|\$'_p\rangle$ give rise to the same probability distribution over grid diagrams G.

The subroutine in step 3 of the verification procedure of Section 6.2.2 starts with the input state $|\varphi\rangle$ and applies the following algorithm.[3]

1. Take the input state $|\varphi\rangle$, append an ancilla register that can hold integers $i \in \{0, ..., q_{max}\}$, and prepare the state

   $$|\varphi'\rangle = U\left(|\varphi\rangle|0\rangle\right)$$

   using the unitary U defined above.

[3] Using a construction from [41], it is possible to associate a Hamiltonian with a Markov chain such as ours. It may also be possible to construct a verifier using phase estimation on this Hamiltonian.

2. Append another ancilla register which holds integers $s \in \{1, ...|\mathcal{S}|\}$ and initialize it to the state $\sum_s \frac{1}{\sqrt{|\mathcal{S}|}}|s\rangle$. This produces the state

$$|\varphi'\rangle \sum_s \frac{1}{\sqrt{|\mathcal{S}|}}|s\rangle$$

on three registers (one that holds grid diagrams G, one that holds integers $i \in \{0, ..., q_{max}\}$ and one that holds integers $s \in \{1, ..., |\mathcal{S}|\}$).

3. Repeat $r = \text{poly}(\bar{D})$ times:

   a) Apply the unitary V, where

$$V = \sum_s \hat{P}_s \otimes |s\rangle\langle s|.$$

   This operator applies the permutation $\hat{P}_s$ to the first two registers, conditioned on the value s in the third register.

   b) Measure the projector

$$Q = I \otimes I \otimes \left(\sum_{s,s'} \frac{1}{|\mathcal{S}|}|s\rangle\langle s'| \right).$$

4. If you obtained the outcome 1 in each of the r repetitions of step 3, then accept the money. In this case apply $U^\dagger$ and then remove the second and third registers. The final state of the first register is the output quantum money state. If you did not obtain the outcome 1 in each of the r iterations in step 3, then reject the money.

For a quantum money state $|\$_p\rangle$ the associated state $|\$'_p\rangle \sum_s \frac{1}{\sqrt{|\mathcal{S}|}}|s\rangle$ prepared in steps 1 and 2 is unchanged by applying the unitary V in step 3a. Measuring the projector Q in step 3b on this state gives $+1$ with certainty. So for good quantum money, all the measurement outcomes obtained in step 4 are $+1$. We can see that good quantum money passes verification.

Now let us consider the result of applying the above procedure to a general state $|\varphi\rangle$. The first step of the algorithm is to prepare $|\varphi'\rangle = U(|\varphi\rangle|0\rangle)$. If a single iteration of the loop in step 3 results in the outcome 1, then the final state of the first two registers is

$$\frac{\frac{1}{|\mathcal{S}|}\sum_s \hat{P}_s|\varphi'\rangle}{\left\|\frac{1}{|\mathcal{S}|}\sum_{s,t} \hat{P}_s|\varphi'\rangle\right\|}.$$

This occurs with probability

$$\left\|\frac{1}{|\mathcal{S}|}\sum_s \hat{P}_s|\varphi'\rangle\right\|^2. \tag{6.4}$$

The entire procedure repeats the loop r times and the probability of obtaining all outcomes 1 is

$$\left\| \left(\frac{1}{|\mathcal{S}|} \sum_s \hat{P}_s \right)^r |\varphi'\rangle \right\|^2,$$

in which case the final state (on the first two registers) is

$$\frac{\left(\frac{1}{|\mathcal{S}|} \sum_s \hat{P}_s \right)^r |\varphi'\rangle}{\left\| \left(\frac{1}{|\mathcal{S}|} \sum_s \hat{P}_s \right)^r |\varphi'\rangle \right\|}.$$

To get some intuition about what states might pass even the first iteration of this test with reasonable probability (6.4), note that the state $\frac{1}{|\mathcal{S}|} \sum_s \hat{P}_s |\varphi'\rangle$ can only have norm close to 1 if most of its terms add coherently. In other words, most of the $\hat{P}_s|\varphi'\rangle$ must be nearly the same (they are all exactly the same for good quantum money).

Since $\frac{1}{|\mathcal{S}|} \sum_s \hat{P}_s = \hat{B}$ is our Markov matrix, the set of states that pass all the rounds is directly related to the mixing properties of the Markov chain – these states correspond to eigenvectors of $\hat{B}$ with eigenvalues close to 1.

6.2.3. Security of the money scheme

We have shown that the quantum money states $|\$_p\rangle$ can be efficiently generated and pass verification with high probability. In this section we discuss why we believe this quantum money is hard to forge. We consider four possible types of attack against our quantum money.

First, an attacker might measure a valid quantum money state $|\$_p\rangle$ to learn a grid diagram with Alexander polynomial p and then generate a superposition containing that diagram that passes verification. One such state is the (correctly weighted) superposition over grid diagrams equivalent to the measured diagram. If an attacker could do this, the attacker's algorithm could be used to solve grid diagram equivalence, i. e. the problem of telling whether or not two grid diagrams represent the same link. This is believed to be a hard problem on average, even for quantum computers. In fact, even deciding whether or not a grid diagram represents the unknot is conjectured to be hard. By the theorem in Section 5.5, a generic oracle attack of this type is most likely impossible.

Second, there are likely to exist grid diagrams of dimension $2\bar{D}$ (our maximum grid dimension) where no allowed grid move reduces the dimension (this follows from the fact that every link has a minimum dimension of grid diagrams that represent it). Because we disallow moves that increase the dimension above $2\bar{D}$, the Markov chain starting from one of these grid diagrams with dimension $2\bar{D}$ will only mix over a small set of diagrams with the same dimension. Uniform superpositions over these sets will pass step 3 of the verification procedure. Step 2 of the verification procedure is designed to reject such superpositions.

Third, if the Markov chain does not mix well, then there will be eigenstates of the matrix $\hat{B}$ with eigenvalues near +1. For such eigenstates there may be counterfeit quantum money states that will pass verification with non-negligible probability. We do not know if these states exist, and even if they do we do not know how to make them. In fact even the simpler question, of finding two equivalent diagrams that require a super-polynomial number of grid moves to go from one to the other (or proving such diagrams do not exist) seems hard.[4]

Fourth, the attacker could use a valid money state $|\$_p\rangle$ and attempt to generate $|\$_p\rangle \otimes |\$_p\rangle$ (or some entangled state on two registers where each register would pass verification). Such an attack worked to forge product state quantum money using quantum state restoration [5]. However, in that case the attack works because the product state money has an embedded classical secret that the attacker can learn to forge the state. In the present work, there is no obvious secret hidden in the money or in the verification algorithm that an attacker could use.

The list above comprises all the lines of attack we were able to think of.

6.2.4. Why not quantum money from graphs?

A similar but simpler quantum money scheme could be designed using graphs instead of knots. Such a scheme would be insecure because, in practice, it is easy to find the isomorphism relating most pairs of isomorphic graphs. The knot based quantum money is an analog of this scheme; we expect it to be secure because knot equivalence is believed to be difficult on average.

In the graph based quantum money scheme, the Hilbert space has basis vectors that encode adjacency matrices of graphs on n vertices. The mint generates the state

$$\sum_{\text{Adjacency matrices } A} |A\rangle|0\rangle.$$

The mint then computes the eigenvalues of the adjacency matrix into the second register and measures the second register to obtain a particular spectrum $R = \{\lambda_1, ...\lambda_n\}$. The resulting state is

$$|\$_R\rangle|R\rangle = \sum_{A \text{ with spectrum } R} |A\rangle|R\rangle.$$

The quantum money is the state $|\$_R\rangle$ and the serial number encodes the spectrum R. The verification procedure is based on a classical Markov chain that, starting from a given adjacency matrix A, mixes to the uniform distribution over adjacency matrices A' that represent graphs isomorphic to A.

Given two adjacency matrices A_0 and $\sigma A_0 \sigma^{-1}$ that are related by the permutation $\sigma \in S_n$, the forger can usually efficiently find the permutation σ (we assume

[4] Hass and Nowik [42] recently gave the first example of a family of knots that require a quadratic number of Reidemeister moves to untangle. Previous lower bounds were only linear.

for simplicity that A_0 has no automorphisms so this permutation is unique). We now show how this allows a counterfeiter to forge the graph based quantum money. The forger first measures the state $|\$_R\rangle$ in the computational basis, obtaining a particular adjacency matrix A with the spectrum R. Starting from this state, the forger adjoins two additional registers (one that holds permutations and one that holds adjacency matrices) and, applying a unitary transformation, the forger can produce the state

$$\sum_{\pi \in S_n} |A\rangle|\pi\rangle|\pi A \pi^{-1}\rangle.$$

Now the forger can use the procedure that solves graph isomorphism to uncompute the permutation held in the second register, producing the state

$$|A\rangle|0\rangle \sum_{\pi \in S_n} |\pi A \pi^{-1}\rangle.$$

The state of the third register will pass verification. The forger can repeat this procedure using the same adjacency matrix A to produce many copies of this state, all of which will appear to be valid quantum money.

Quantum money from knots appears invulnerable to this attack. Given two grid diagrams G_1 and G_2 that represent the same link, we believe it is hard (on average) to find a sequence of grid moves which transform G_1 into G_2. Even given an oracle for the decision problem of determining whether two links are equivalent, we do not know how to find a sequence of Reidemeister moves relating the links. We hope this discussion has motivated some of the choices we have made for our knot based quantum money.

6.3. Details of the Markov chain on planar grid diagrams

The Markov chain in section 6.2.2.1 acts on the configuration space of pairs (G, i) where $1 \leq i \leq q(d(G))$. Here we describe an algorithm to implement this Markov chain. For each update of the state of the Markov chain, we first choose several uniformly random parameters:

$$\begin{aligned} j: &\quad \text{an integer from 1 to 8} \\ w: &\quad \text{an integer from 0 to } q_{max}^2 \\ x, y: &\quad \text{integers from 0 to } 2\bar{D} - 1 \\ k: &\quad \text{an integer from 0 to 3} \end{aligned}$$

where q_{max} is the maximum value attained by the function $q(d)$ for $d \in \{2, ..., 2\bar{D}\}$. We then update the configuration (G, i) in a manner that depends on the values of these variables, where j picks the type of update being performed, and the other variables are used as parameters:

- If $j = 1$, set
$$i \leftarrow (i + w) \mod q(d(G))$$
Leave G unchanged. This is the only move which changes i, and it is used to mix between different labels.

- If $j = 2$, cyclically permute the columns of G by moving each column to the right by one. Leave i unchanged.

- If $j = 3$, cyclically permute columns of G by moving each column to the left by one. Leave i unchanged.

- If $j = 4$, cyclically permute the rows of G by moving each row up by one. Leave i unchanged.

- If $j = 5$, cyclically permute the rows of G by moving each row down by one. Leave i unchanged.

- If $j = 6$ and $x + 1 < d(G)$, then check whether transposing columns x and $x + 1$ is a legal move (as defined in section 6.1.3). If so, transpose them; otherwise, do nothing. Leave i unchanged.

- If $j = 7$ and $y+1 < d(G)$, then check whether transposing rows y and $y+1$ is a legal move (as defined in section 6.1.3). If so, transpose them; otherwise, do nothing. Leave i unchanged.

- If $j = 8$, $k = 0$, $x, y < d(G)$, and there is a marker (X or O) at position (x, y), then consider stabilizing by adding markers forming an L to the upper right of position (x, y). In this case, construct G' from G by adding a new column to the right of x and a new row above y, deleting the original marker, adding markers of the same type at $(x + 1, y)$ and $(x, y + 1)$, and adding a marker of the opposite type at $(x + 1, y + 1)$. Then if $i \leq q(d(G'))$, set $(G, i) \leftarrow (G', i)$. If not, leave the configuration unchanged.

- If $j = 8$, $k = 0$, $x, y < d(G) - 1$, there is no marker at (x, y), and there are markers at positions $(x + 1, y + 1)$, $(x + 1, y)$, $(x, y + 1)$, then consider destabilizing by removing markers forming an L to the upper right of position (x, y). In this case, construct G' from G by removing column $x + 1$ and row $y + 1$ (thus removing those three markers) and adding a new marker of the appropriate type at (x, y) (measured after deleting the row and column). If $i \leq q(d(G'))$, set $(G, i) \leftarrow (G', i)$. If not, leave the configuration unchanged.

- If $j = 8$, and $k \in 1, 2, 3$, rotate the grid diagram by $90k$ degrees counterclockwise and then apply the update rule corresponding to $j = 8, k = 0$ with the same (x, y). After applying the update, rotate the diagram back by the same amount.

The parameter j determines the type of move. The first move ($j = 1$) is used to permute between different labels of the same graph; it is the only move that changes the labels. Moves 2–5 are cyclic permutations of rows and columns, while moves 6 and 7 are transpositions. Finally, $j = 8$ stabilizes or destabilizes an L in a direction (⌝ ⌟ ⌞ or ⌜) depending on k (0, 1, 2, or 3 respectively). In all the moves x is treated as column index (which describes where to apply the move), and y is treated as a row index.

For fixed (j, w, x, y, k), the update of the state is an easily computable, invertible function (in other words it is a permutation of the configuration space). This is easy to see for any of the moves with $j \in \{1, ..., 7\}$. One can check that each move with $j = 8$ is its own inverse. This justifies our assertion that the Markov matrix B can be written in the form of equation 6.3.

In the notation of Section 6.2.2, $s = (j, w, x, y, k)$ and

$$\begin{aligned}\mathcal{S} =& \{1, \ldots, 8\} \times \{0, \ldots, q_{\max}^2\} \times \\ & \{0, \ldots, 2\bar{D} - 1\} \times \{0, \ldots, 2\bar{D} - 1\} \times \{0, 1, 2, 3\}.\end{aligned}$$

7. Sealed states and quantum blackmail

Imagine that you want to blackmail someone with incriminating documents. You are worried that your victim might kill you instead of paying up; for insurance, you give a copy of your incriminating documents to your attorney to be unsealed and published in the event of your untimely death. You meet with your victim and extort some quantum money.

Of course, your reputation as an honest blackmailer would be destroyed if the incriminating documents got out even though your victim paid up. To be safe, you ask your attorney for the documents back.

Is there any way your attorney can prove that he or she did not open and copy the documents before returning them?

With only classical techniques, the best you can do is to physically seal the documents before giving them to your attorney and check that the seal is intact when you get the documents back. If your attorney is good at covertly opening envelopes or if you want to send the documents to your attorney over the internet instead of in person, this is not good enough. Encrypting the documents would not help because your attorney needs to be able to read them if you die.

By the no-cloning theorem, it is conceivable that you could encode the documents into a *sealed state* and give a subsystem of that state to your attorney. He or she cannot directly copy the state, so perhaps any attempt to read the state would be detectable.

A quantum blackmail protocol has two players: Belinda the blackmailer and Charlie the co-conspiring attorney. Belinda has some secret message and, from that message, produces a state on two registers, B and C. She gives register C to Charlie. Charlie can do one of three things:

- He can *unseal* the state to learn the message.
- He can give the state in register C back to Belinda and tell her that he did not unseal the message.
- He can cheat: he performs a local operation on the state in register C, gives that state to Belinda, and tells her that he did not unseal the state.

If Charlie unseals the state, then the protocol is done. If he tells Belinda that he did not unseal the message, then Belinda will make a measurement to decide whether to believe him. If Charlie did not cheat, then Belinda believes him with probability $1 - \varepsilon_c$, where ε_c is the protocol's completeness error. If Charlie does cheat, then

Belinda may or may not catch him and he gains some advantage in determining what the sealed message was.

In the straightforward case, the message being sealed is a single classical message, and Charlie must be able to learn the message with some probability p if he unseals the state. If he cheats, then Belinda should catch him with high probability. This turns out to be impossible to achieve with good security: in quantum computing, any measurement that has a nearly deterministic outcome can be done with almost no damage to the state. If p is large, Charlie can learn the message with minimal damage to his state and is unlikely to be caught. If p is small, the protocol is not very useful, and Charlie's chance of getting away with cheating does not decrease rapidly as p decreases. In particular, Charlie can always recover the message with probability p, and Belinda will catch him with at most probability $\varepsilon_c + \sqrt{1-p}$. Charlie's cheating strategy does not require difficult computation. The proof is given in Section 7.1.

Nevertheless, there are at least two ways to achieve secure quantum blackmail protocols.

The simpler case is when Belinda has a collection of distinct pieces of classical information and unsealing the state only needs to reveal one of them to Charlie. This is realistic: if Belinda has several compromising pictures of her victim, then it may be sufficient for Charlie to have access to a single random picture as insurance. Belinda can put a random picture in the C register and keep the purification of the state in the B register; if Charlie copies the picture, he will know which picture it was and break the entanglement between the registers. Section 7.2 discusses the security properties of this protocol.

The more complicated case uses a new type of quantum computing resource: a computation that cannot be done on a quantum computer. The generic attack against sealed states relies on *reversibly* unsealing the state. If part of the unsealing process cannot be done on a quantum computer, then the attack may fail. Of course, any calculation that can be done on a classical computer can, in principle, be done on a quantum computer as well. Until someone invents perfect artificial intelligence, however, there will be calculations that can be done by humans but not by any computer. Since humans are not coherent quantum computers, we can use such a calculation to implement general-purpose sealed states. The idea is to encode a message as a superposition of many different images or sounds, any of which can be decoded by a human but not by a computer. Since humans cannot be run in reverse, showing the image or sound to a human will break the superposition. Section 7.3 gives an example of this type of protocol.

7.1. A bound on soundness and completeness

Suppose that Belinda has a single classical message m. She prepares a sealed message in register C and gives it to Charlie. Without loss of generality, we

assume that Belinda keeps a purification of Charlie's state in register B, so the full sealed state is $|\psi_m\rangle_{BC}$. Charlie knows some algorithm that takes register C as input and outputs m in polynomial time with probability p.

If Charlie returns register C unmodified to Belinda, then Belinda will perform some measurement on registers B and C to determine whether Charlie cheated. If Charlie did not cheat, then Belinda will believe him with probability $1 - \varepsilon_c$, where ε_c is the completeness error of the protocol.

Charlie can try cheat. For example, he could perform some measurement on register C to try to learn m and give the state that remains in register C back to Belinda. If he tells her that he did not look at the message, she will believe him with probability $1 - s$, where s is the soundness of the protocol. In general, s can depend on Charlie's cheating strategy.

Claim. Charlie has a strategy that recovers m with probability p such that Belinda will catch him with probability $s \leq \varepsilon_c + \sqrt{1-p}$.

Proof. If Charlie were to open the message instead of cheating, he would perform some measurement. Without loss of generality, that measurement consists of a unitary operator U followed by a projective measurement $\{P_i\}$, both on register C. The projective measurement has one outcome per possible message. If Charlie needs any ancillas for his measurement, we can absorb them into register C.

Charlie's cheating strategy is based on the measurement he uses to open the state. Charlie acts only on register A, but we will keep track of the joint state of both registers. The initial state is $|\psi\rangle_{BC}$. Charlie applies $\mathbb{I} \otimes U$ and then makes the projective measurement $\{P_i\}$. With probability p, Charlie gets the outcome corresponding to m and learns the state. With probability $1 - p$, Charlie gets a different outcome and does not learn the state. In either case, Charlie applies $\mathbb{I} \otimes U^\dagger$ to the state that remains after the measurement and gives that state to Belinda.

If Charlie gets the correct outcome m, the final state of registers B and C is

$$|\psi'\rangle = \frac{1}{\sqrt{p}} \left(\mathbb{I} \otimes U^\dagger\right) (\mathbb{I} \otimes P_m) (\mathbb{I} \otimes U) |\psi\rangle.$$

The operator $\left(\mathbb{I} \otimes U^\dagger\right) (\mathbb{I} \otimes P_m) (\mathbb{I} \otimes U)$ is a projector. Let

$$|\varphi\rangle = \left(\mathbb{I} \otimes U^\dagger\right) (\mathbb{I} \otimes P_m) (\mathbb{I} \otimes U) |\psi\rangle = \sqrt{p}|\psi'\rangle \text{ and}$$
$$|\varphi^\perp\rangle = \left[\mathbb{I} - \left(\mathbb{I} \otimes U^\dagger\right) (\mathbb{I} \otimes P_m) (\mathbb{I} \otimes U)\right] |\psi\rangle.$$

Then the trace distance between the initial and final state is

$$\begin{aligned}
&D\left(|\psi\rangle\langle\psi|, \frac{1}{p}|\varphi\rangle\langle\varphi|\right) \\
=&D\left(|\varphi\rangle\langle\varphi| + |\varphi^\perp\rangle\langle\varphi^\perp| + |\varphi^\perp\rangle\langle\varphi| + |\varphi\rangle\langle\varphi^\perp|, \frac{1}{p}|\varphi\rangle\langle\varphi|\right) \\
=&\frac{1}{2}\operatorname{Tr}\left|\begin{array}{cc} p-1 & \sqrt{p\,(1-p)} \\ \sqrt{p\,(1-p)} & 1-p \end{array}\right| \\
=&\sqrt{(1-p)^2 + p\,(1-p)} \\
=&\sqrt{1-p}
\end{aligned}$$

The trace distance is an upper bound on the difference between the probability that Belinda accepts $|\psi\rangle$ and the probability that Belinda accepts $|\varphi\rangle$. So $s - \varepsilon_c \leq \sqrt{1-p}$. □

Even this bound is difficult to achieve. The obvious protocol is for Belinda to send Charlie the C register of

$$|\psi_m\rangle_{BC} = \frac{1}{\sqrt{2}}\left[|0\rangle_B|0\rangle_C + |m\rangle_B|m\rangle_C\right],$$

where $|0\rangle$ is an arbitrary nonsense state. If Charlie returns the state unopened, Belinda verifies by projecting onto $|\psi_m\rangle$. This protocol is complete (i.e. $\varepsilon_c = 0$), and it is reasonably sound against simple attacks. If Charlie measures his qubit in the computational basis, Belinda will detect his cheating with probability ½, which is worse than the bound of 71%.

Charlie has other options, though. For example, if he is worried that Belinda is blackmailing his friend, then he can verify that m is *not* a compromising message about his friend. More generally, Charlie can choose any classical function $g(m) \in \{0, 1\}$ that satisfies $g(0) = 0$. Then Charlie can measure g on his register. If $g(m) = 1$ (e.g. m is a compromising message about Charlie's friend), then Charlie gets the correct answer w.p. ½ and gets caught w.p. ½. But if $g(m) = 0$, then Charlie gets the correct answer w.p. 1, and Belinda will not catch Charlie.

This weakness of quantum blackmail is consequence of the fact that quantum mechanics allows Charlie to make any measurement with a definite outcome without causing any damage to the state being measured. To work around this weakness, we need an extra assumption.

7.2. Multiple compromising pictures

If Belinda has more than one message and she is willing to give Charlie a sealed state that can only reveal one of them, then she can do better. For example, suppose she has n compromising pictures $m_1, \ldots, m_n$ of her victim, and she considers the

threat of any one of them being published to be adequate to ensure her safety. Belinda is therefore generates a quantum random selection of one picture

$$|\psi\rangle_{BC} = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle|m_i\rangle$$

and sends register C to Charlie. Of course, any tabloid would take great pains to authenticate a photograph before publishing it, so we will assume that Charlie has no interest in cheating unless he can recover an entire intact photograph. If can do this, then he knows which photograph Belinda sent him, and register B collapses to a single value of i. No matter what state Charlie sends Belinda, Belinda can detect that he cheated with probability at least $\frac{n-1}{n}$. As n becomes large, this protocol becomes more secure.

This protocol is vulnerable to other attacks, though. Charlie could measure a property that all n pictures have in common without damaging the state at all. For example, he could use a quantum image recognition program to figure out who is in the pictures.

To prevent this type of attack and allow exponential security, we need a new type of resource.

7.3. CAPTCHAs

The fundamental weakness of quantum cryptography that makes quantum blackmail difficult is that any computation can be performed coherently. If Charlie can calculate something based on a sealed state that Belinda gives him, then he can do the same calculation coherently, measure some property of the result, and undo the original calculation. If the unsealing process required that Charlie does something that cannot be done on a quantum computer, then we could block these attacks. If quantum computers ever become as powerful as classical computers, then the only things that quantum computers will not be able to do are things that no computer at all can do.

CAPTCHAs, a type of spam-preventing technology on the internet, are based on a computation that can only be reliably done by humans. A CAPTCHA is a "Completely Automated Public Turing test to tell Computers and Humans Apart" [43]. To use a CAPTCHA, a website generates a random word or number x. The website then computes a picture, sound, or other message based on x that is meant to be decoded by a human. Any human should be able to find x by inspecting the web page, but no polynomial-time algorithm should be able to find x with non-negligible probability.

A CAPTCHA is defined as a communication protocol, possibly with multiple rounds. We need a function instead of a communication protocol, and we need the function to be secure against quantum adversaries. We therefore propose the following definition:

Definition 11. An human-invertible one-way function is a one-to-one function $f : \{0,1\}^* \to I$ that maps strings of bits to a set I and has these properties:

- The function f can be evaluated in quantum polynomial time.
- Given $f(x)$ for unknown x, a human acts as an oracle that computes x in one query with negligible probability of error. This process is incoherent: $f(x)$ leaks to the environment. Most likely, I will be a set of images, sounds, or other media that a human can look at.
- There is no polynomial-time quantum algorithm that can compute f^{-1} with non-negligible probability of success.

Standard CAPTCHAs are randomized, but we will assume that f can derive any randomness it needs from its argument.

A human-invertible one-way function is very similar to a trapdoor one-way function; the difference is that instead of being invertible using trap door information, it is invertible by asking a human to invert it. Without access to a human, it works just like a trapdoor one-way function with unknown trap door information. We will therefore construct a cipher from f in much the same way that public-key ciphers are constructed using trap door functions. The standard construction is Bellare and Rogaway's OAEP [44], and we will use a similar construction. OAEP is randomized, and we will replace the randomness with entanglement. To unseal a message, Charlie must show a particular value of f to a human, breaking the entanglement in the process.

Let n be the length of the message being sealed. Following OAEP, choose security parameters k and k_0, where $n = k - k_0$. Let $G : \{0,1\}^{k_0} \to \{0,1\}^n$ be a pseudorandom generator and $H : \{0,1\}^n \to \{0,1\}^{k_0}$ be an ideal hash function. Both G and H are publicly known. The sealed state encoding y is

$$|\psi_y\rangle_{BC} = \frac{1}{\sqrt{2^{k_0}}} \sum_{r \in \{0,1\}^{k_0}} |r\rangle_B |\mathcal{E}_r^{G,H}(y)\rangle_C,$$

where

$$\mathcal{E}_r^{G,H}(y) = f(y \oplus G(r) \parallel r \oplus H(y \oplus G(r))$$

is the OAEP encrypted version of y. The $\parallel$ operator denotes concatenation of strings of bits, and $\oplus$ is bitwise exclusive or.

To seal a message y, Belinda generates $|\psi_y\rangle_{BC}$ and gives Charlie the C register. To unseal the message, Charlie measures register C and shows it to a human. The human inverts f to recover $y \oplus G(r) \parallel r \oplus H(y \oplus G(r))$ for some unknown value of r. Charlie then computes $H(y \oplus G(r))$ to recover r, $G(r)$, and y.

If Charlie does not unseal the message, he returns register C to Belinda, who measures $|\psi_y\rangle\langle\psi_y|$ on the combined state in registers B and C. If the outcome is 1, then she believes Charlie; if not, she accuses Charlie of cheating. This protocol

is fully complete – if Charlie does not cheat, then Belinda will believe him with probability one.

If Charlie cheats, he can use an arbitrarily complicated strategy to select the values of f that he asks a human to invert. Let $Q \subseteq I$ be the set of all f values that he will ever ask a human to invert. Of course, Charlie can ask a human to evaluate f^{-1} even after Belinda has made her measurement, so Belinda cannot possibly know the set Q. Nonetheless, Charlie is constrained to make a polynomial number of queries, so $|Q| = O(\text{poly}(k, k_0))$. Let

$$R = \left\{ r : \mathcal{E}_r^{G,H}(y) \in Q \right\}$$

and define projectors

$$T = |\psi_y\rangle\langle\psi_y| = \frac{1}{2^{k_0}} \sum_{r,r' \in \{0,1\}^{k_0}} |r\rangle_B |\mathcal{E}_r^{G,H}(y)\rangle_{CC} \langle \mathcal{E}_{r'}^{G,H}(y)|_B \langle r'| \text{ and}$$

$$U = \frac{1}{2^{k_0} - |R|} \sum_{r,r' \in \{0,1\}^{k_0} \setminus R} |r\rangle_B |\mathcal{E}_r^{G,H}(y)\rangle_{CC} \langle \mathcal{E}_{r'}^{G,H}(y)|_B \langle r'|.$$

Belinda's measurement is T, the projector onto the uniform superposition of all r values, each paired with its corresponding $\mathcal{E}_r$ value. The projector U is almost the same; it projects onto the uniform superposition of r values that are useless to Charlie – this is the set of r values for which Charlie will never ask a human to invert $f(\mathcal{E}_r^{G,H})$. Belinda cannot measure U as she does not know the set R, but if she could and the outcome of measuring U were 1, then Charlie could only attempt to decrypt $f(\mathcal{E}_r^{G,H})$ without using a human to evaluate $f^{-1}(f(\mathcal{E}_r^{G,H}))$.

Belinda's measurement T and the operator U are both rank-1 projectors, and they project onto states that differ negligibly from each other. If Belinda obtains the outcome 1 from T and therefore believes Charlie, then either she got unlucky due to the difference between T and U, which occurs with negligible probability, or Charlie's queries are all useless.[1]

In the event that Charlie's queries are useless, then recovering x is mostly equivalent to breaking OAEP, which should be impossible as long as the human-invertible one-way function f is secure. The standard OAEP security proof [44] fails in this context for two reasons: it assumes a classical adversary, and it assumes that the trapdoor function is a permutation; f is merely one-way. The latter problem should be easily correctable, but the former will be more challenging. We are unaware of any meaningful security proofs of OAEP (or, for that matter, any other public-key cipher construction) against quantum adversaries.

[1] This is not the same thing as saying that if Belinda believes Charlie, then Charlie's queries are useless with high probability. Charlie could, for example, unseal the message and give Belinda the classical state $|\mathcal{E}_r^{G,H}(y)\rangle$ in register C. Belinda will believe him with negligible probability, but if she believes him then his queries are still useful with probability 1.

This protocol also is also somewhat resistant to attacks that break the human-invertible one-way function after Belinda makes her measurement – if Belinda's test passes with non-negligible probability, then Charlie does not know anything that would let him reliably determine the value of r. It is hard to imagine that such any quantum state he generated from register C without being able to invert f would let him later recover y even with unlimited computational power.

If Belinda wants to seal a very long message, it could be more efficient to generate a short random key, encrypt the message against the key, and seal the key instead of the message.

7.4. Open questions

These protocols require Belinda to store a quantum state that is entangled with the sealed state. It should be possible to eliminate the entanglement. One naive approach is to eliminate register B, making the sealed state

$$\frac{1}{\sqrt{2^{k_0}}} \sum_{r \in \{0,1\}^{k_0}} |\mathcal{E}_r^{G,H}(y)\rangle.$$

This is neither practical nor secure: preparing this state is likely as hard as index erasure [45], and if Charlie unseals the message then he can recreate the original state with whatever algorithm Belinda used to prepare it in the first place. The difficulty in preparing the state can be resolved by giving Charlie both registers, that is

$$\frac{1}{\sqrt{2^{k_0}}} \sum_{r \in \{0,1\}^{k_0}} |r\rangle|\mathcal{E}_r^{G,H}(y)\rangle.$$

This is insecure for the same reason. An improved version would be

$$\frac{1}{\sqrt{2^{k_0}}} \sum_{r \in \{0,1\}^{k_0}} e^{i\varphi(r)}|r\rangle|\mathcal{E}_r^{G,H}(y)\rangle,$$

where the function $\varphi(r)$ is a secret known only to Belinda. This appears to be secure, although the security proof would be more complicated.

Regardless of whether Belinda needs to store a quantum state, all of these protocols involve giving Charlie a highly entangled state. A protocol in which Charlie's state was a tensor product of a large number of low-dimension systems would be very interesting, since it could be built with much simpler quantum computing technology. Designing such a protocol that is secure even if Charlie can make entangling measurements may be difficult.

Putting aside quantum blackmail in particular, the technology for analysing the security of even classical cryptographic constructions against quantum attack is limited. There is extensive literature on the security of constructions as varied as

the Luby-Rackoff block cipher, psuedorandom functions, public-key cryptosystems, and modern block cipher modes. Many of these are provably secure against classical attack assuming that some underlying primitive is secure. Little progress has been made in defining security against quantum attacks. Classical attacks can take many forms (e.g. known-plaintext attacks, adaptive chosen-ciphertext attacks, etc.); this taxonomy will need to be extended to meaningfully discuss security in a post-quantum world (e.g. what is a non-adaptive chosen-quantum-ciphertext attack, and when is it relevant?). Even less progress has been made in proving the security of cryptographic constructions against quantum attack; the best we can say is that no one has found generic attacks better than Grover's algorithm.

Until the basic technology for analyzing the security of constructions like OAEP against quantum attacks is in place, it will be difficult to make rigorous statements about the security of even straightforward quantum cryptographic constructions like quantum blackmail.

Part II.

Radio astronomy

8. Introduction

Fourteen billion years ago [46], the Big Bang filled the universe with hot, dense, glowing plasma. About four hundred thousand years later, the plasma cooled off enough to turn into neutral hydrogen. Neutral hydrogen is transparent, and it filled the universe for several hundred million years as the first stars formed. Those stars emitted ultraviolet light and ionized the neutral hydrogen around them, eventually filling most of the universe with plasma again. This plasma, unlike the much denser plasma from the Big Bang, was mostly transparent. The period over which the neutral hydrogen formed is called the *epoch of reionization* or EoR.

Most of what we know about the history of our universe comes from things we can observe with telescopes. The universe is a big place and, given the scale of the universe, light takes a long time to cross it. If we look far away, we can see back in time – we see each object in the sky as it was when its light was emitted. If we look far enough back, we can still see the edge of the glowing plasma from the big bang. That plasma was opaque, and we cannot see anything through it. The edge of the plasma is called the *surface of last scattering* and its glow is the *cosmic microwave background* or CMB. Everything, even light, cools as the universe expands, so after fourteen billion years, the cosmic microwave background is a chilly 2.7 Kelvin.

Between the surface of last scattering and the EoR, there is very little that we can see. Neutral hydrogen gas is almost completely transparent, and there was nothing else in the universe back then that we can see today. But, if we could somehow see the neutral hydrogen gas and measure its distribution, we would learn an enormous amount about the history of the universe and the process of reionization.

Fortunately, neutral hydrogen is not completely transparent. It can undergo a hyperfine transition and interact with light at a wavelength of 21 cm. The universe is expanding, and over time all of the light in the universe is redshifted; that is, its wavelength increases. As a result, the neutral hydrogen in the universe modulated the intensity of light from the CMB that had a wavelength of 21 cm *when it interacted with the hydrogen.* As in Figure 8.1, the light that interacted with hydrogen closer to us had less time to redshift on its way to us and therefore has a shorter wavelength now than light that interacted with hydrogen farther away. If we could measure the spectrum of this light, we could make a three-dimensional map of the temperature of the neutral hydrogen in the early universe. Using such data, we could learn about the formation of the first stars and the shape of the early universe in unprecedented detail.

The 21 cm line from the EoR has never been detected, let alone measured. By the

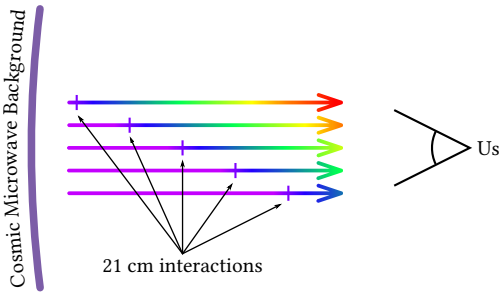


Figure 8.1: Light that interacts with hydrogen gas far away is redder than light that interacts closer to us.

time the light modulated by neutral hydrogen in the EoR reaches us, it has been redshifted to frequencies between 30 and 200 MHz. The modulation of the light corresponds to a shift in the effective temperature of the sky at these frequencies on the order of 30 mK. This very faint signal is hidden behind nearby radio sources that can be two or three orders of magnitude brighter. In order to detect the faint EoR signal, a telescope will need to take pictures of the sky at these frequencies with greater precision and sensitivity than is possible with any existing instrument.

Building a single-dish radio telescope large enough to measure the EoR signal is essentially impossible. Instead, astronomers use large arrays of small antennas to simulate a larger radio telescope than could be built directly. This technique is called aperture synthesis, because the telescope synthesizes the image that it would have seen if it were a single radio dish with a large aperture.

A telescope large enough to measure the EoR signal will contain a very large number of small antennas. It will collect data from each antenna, digitize that data, and process it in real-time to form images of the sky. The more antennas there are, the more complex the calculations become.

Throughout this thesis, I use the number N to represent the number of antennas that compose a radio telescope. Traditional aperture synthesis techniques require computing power proportional to N^2 to process the data from N antennas. Even though computing power seems cheap today, in a large enough telescope, the hardware needed to perform this computation costs more than anything else, so the number of antennas in most designs is limited by the cost of computing power.

With the FFT telescope group at MIT, I worked on a design for a radio telescope that operates between 120 and 200 MHz and will scale to extremely large N. We use a different aperture synthesis technique based on Fast Fourier transforms [47] with computational costs proportional to $N \log N$ instead of N^2. This eliminates the cost of computers as the main limit on the number of antennas that can be combined into a radio telescope, and it changes many of the other factors in the design of the array. In this type of telescope, the cost of each antenna matters regardless of how large the telescope becomes, so we focus on reducing the cost

of each antenna as much as possible. To detect the EoR signal, we will need hundreds of antennas observing a large portion of the sky for many months [48]. To measure the signal precisely, we will eventually want thousands if not hundreds of thousands of antennas.

8.1. Antenna basics

Before we get into the details of how to measure the 21 cm signal from the EoR, it will be useful to cover some of the basics of radio astronomy and engineering.

There are a few ways to measure radio signals coming from the sky. One way is to build a detector that measures incident power and then construct optics to focus energy onto the detector. For example, our eyes use chemical sensors, optical telescopes use semiconductor sensors, X-ray telescopes use a variety of energy-detecting devices, and microwave detectors often use bolometers. These devices have the downside that their optics must be made larger to increase their sensitivity and resolution, and we want to collect light from a much larger area than any dish we could conceivably build. To avoid this limitation, we can instead use antennas to collect and measure radio energy with phase information intact. We can then use an array of antennas to make pictures of the sky.

Radio astronomers think in an unusual set of units. We describe the brightness of a patch of sky in terms of its *brightness temperature*: an area with brightness temperature T at some frequency emits light of the same intensity as a blackbody source with that temperature. The corresponding *intensity* or *spectral radiance* I_ν is the power per unit frequency per steradian of solid angle in the sky. Spectral radiance and brightness temperature are related by Planck's law

$$I_\nu = \frac{2h\nu^3}{c^2}\frac{1}{e^{\frac{h\nu}{kT}} - 1}.$$

At the low frequencies we care about (around 150MHz), $h\nu \approx 10^{-25}$ J. At 100K, $kT \approx 1.4 \cdot 10^{-21}$ J, and brightness temperature of the sky in our frequency band does not go far below 100 K,[1] so we can take $\frac{h\nu}{kT} \ll 1$. This gives the Rayleigh-Jeans approximation

$$I_\nu \approx \frac{2\nu^2 kT}{c^2}.$$

A source at room temperature has an intensity of around $2 \cdot 10^{-21}$ W $\cdot$ m^{-2} steradian^{-1}Hz^{-1}, and the fluctuations that we are looking for (20 mK) are on the order of 10^{-25} W $\cdot$ m^{-2}steradian^{-1}Hz^{-1}.

When describing electrical signals on a wire, it is easier to think about the power or power spectral density of the signal than the voltage or current. In practice,

[1] The CMB has a brightness temperature of 2.7 K, but there are many other sources of radio waves below 200 MHz that make the effective temperature of the sky much higher. Some of the major sources are the Galactic synchrotron and the Sun.

powers vary over a wide range, so we often use decibels relative to some reference. The expression 0 dBm means zero decibels above one milliwatt, so 0 dBm $= 1$ mW. Similarly, 30 dBm $= 1$ W and -30 dBm $= 1$ μW.

Almost everything emits thermal noise (i.e. Johnson-Nyquist noise) onto any attached wire. At low frequencies, this noise has power $k_B T$ per unit frequency, where T is the temperature of the noise source. At 300K, this is -174 dBm/Hz. The spectrum of thermal noise is different in space and on a wire. In space, a thermal source has power spectral density proportional to ν^2; in a wire, a thermal source has a flat spectrum. The difference is due to the number of dimensions: space is three-dimensional and wires are one-dimensional.

An antenna is a device used to collect incoming radiation of some polarization and turn it into a signal on a wire. There are a huge variety of antennas, differing in their efficiency, bandwidth, size, collecting area, directivity, and cost. Antennas are linear devices: when an antenna receives radiation at a wavelength of λ with intensity I from some direction (θ, φ) over a solid angle $d\Omega$, the incoming power per unit area is $I d\Omega$. The power that the antenna outputs onto the attached wire is $P_{\text{out}} = A_{\text{eff}}(\lambda, \theta, \varphi)\, I d\Omega$. The factor A_{eff} has units of area and is called the effective area of the antenna. The function A_{eff} can have a complicated dependence on angle, especially for larger antennas, but *all* antennas satisfy one basic constraint: the effective area, averaged over the entire sky, is at most $\frac{\lambda^2}{4\pi}$.[2] This constraint only applies to antennas. A large black sphere, for example, absorbs light with an effective area of $4\pi r^2$, but a black sphere turns that absorbed light into heat instead of converting it linearly into a signal on a wire.

The collecting area of a a telescope is the most important parameter that determines its sensitivity to small dim objects, as long as that telescope has a wide enough field of view to see its target. But when designing a survey telescope that looks for a dim signal across a wide area of the sky, the parameter we care about is *étendue*, which is the collecting area integrated over the solid angle observed. This gives us the main design constraint on an instrument to measure the 21 cm EoR signal: each antenna, *no matter how advanced*, contributes an étendue of at most $\frac{\lambda^2}{4\pi}$.

Our goal, then, is to build an instrument with good enough antennas and as many of them as possible. This principle guides our entire design.

8.2. Interferometry

The type of array we are designing, made of many antennas that collectively produce an image of the sky, is called an interferometer.

[2] This follows from the second law of thermodynamics: a perfect antenna immersed in a uniform blackbody field absorbs all incident power coming from the attached wire, just as a resistor on the wire would. It must therefore emit the same amount of power onto the wire as a resistor. An imperfect antenna can reflect some power and emit less.

Imagine monochromatic radio waves coming from a single point in the sky. The radio sources we are looking at are very far away, so the radiation coming from any given point is effectively a plane wave by the time it reaches us. A monochromatic plane wave hitting a single antenna looks like a sine wave, and, assuming the antennas are identical, that plane wave hitting another antenna looks like another sine wave with the same amplitude. If we look at the signal from only one antenna, it could be coming from anywhere, but if we compare the signals from a pair of two antennas, we will find that they have different phases. Light from anywhere other than directly overhead takes a different amount of time to hit each antenna and experiences a different phase lag as a result. This *difference* in phases can tell us where the light came from.

A traditional digital interferometer processes each pair of antennas separately. It correlates the signals from each pair to estimate a function called *visibility*. It calculates these visibilities over a period of time, averages them, and sends them to a computer that turns the visibilities into a picture. This works well and can produce excellent pictures, but it has a serious downside: cost.

Each antenna produces data at some rate. If we collect radio waves from the entire 120 MHz to 200 MHz band simultaneously, we are producing at least 160 million samples (i.e. numbers) per second from *each* antenna. Each of these samples is about one byte of data, so this is 160MB/second. That is not so bad; computers are fast.

But, from these samples, we need to compute a visibility for each *pair* of antennas. If we have N antennas, we have about $\frac{N^2}{2}$ pairs of antennas, and each visibility calculation produces data at about the same rate as an antenna. We can discard unneeded bits to cut this down by maybe a factor of ten, so suppose that each pair of antennas produces visibility data at 16MB/second. That does not sound bad at all, especially since we do not have to store all of that data.

Some day, we would like to build an array with a million antennas, giving us excellent sensitivity to measure the EoR signal. Those million antennas would produce *half a trillion* visibilities at 16MB/second each. That comes out to 8 exabytes or $8 \cdot 10^{18}$ bytes every second. Even though we never have to store all of that data, buying enough computers to compute the numbers in the first place will be absurdly expensive for the foreseeable future.

There are two solutions: either throw away most of the data, making the telescope less powerful or find a different way to make a map that does not require calculating all of the visibilities.

8.3. Fourier transforms

The mathematical trick that allows us to avoid calculating the visibility for each pair of antennas is based on an entirely different way of looking at a telescope. The light coming from the sky is a wave, and there are two natural ways to express

a waveform: in the position basis and in the wave vector basis. These bases are related by a Fourier transform.

In the position basis, we can write down the electric field at any point in space and time. The electric field is a vector field that fills space, and its value at a position with coordinates (x, y, z) and time t is $\vec{E}(x, y, z, t)$. This representation is useful because it corresponds to what we can measure with antennas: an antenna is essentially just a device that measures the electric field at some position and turns it into a voltage on a wire.

In the wave vector basis, we can think of the electric field as being the sum of many modes, each of which is a sine wave that has some frequency ω and has a phase that depends linearly on position. At a position (x, y, z) and time t, a mode $(\vec{k}, \omega)$ has phase $\omega t - k_x x - k_y y - k_z z$. All modes travel at the speed of light, which means that they satisfy the dispersion relation

$$k^2 = \frac{\omega^2}{c^2}.$$

We call the quantity $\vec{k}$ the wave vector of the mode. The wave vector has an important physical significance: it points in the direction that the wave is moving. This means that light with frequency ω from any given patch of the sky corresponds to the electromagnetic wave component with frequency ω and wave vector pointing toward us from that patch of the sky.

These bases for electromagnetic waves are related by a Fourier transform. In fact, an ordinary optical telescope and even a human eye are in effect just devices that Fourier transform incoming light. On a computer, Fast Fourier Transform (FFT) algorithms can convert between bases very efficiently: to convert N evenly-spaced position-basis measurements into the wave vector basis takes $O(N \log N)$ operations. This means that, in principle, we could build a telescope with many antennas on an evenly-spaced grid, measure the electric field as a function of time, and compute the FFT. The resulting amplitudes in the wave vector basis will be a picture of the sky.

In practice, there are plenty of reasons why it is convenient to work with visibilities instead of images. For example, calibrating an interferometer and accounting for the rotation of the earth are both easier and better-understood in terms of visibilities. But we can compute an uncalibrated image of the sky, square the values to compute intensities instead of amplitudes, and Fourier transform back to recover the visibilities without losing any important information [47]. The details of this calculation are in Section 9.3.

With one million antennas, this reduces the computer power needed to build an interferometer by a factor of about one hundred thousand. With some care, the total cost of the telescope can be made almost linear in the number of antennas. If we control the cost per antenna design an easily scalable device, then we can build as large an array as we want for only the incremental cost of its parts.

9. Aperture synthesis and electric field gridding

To show how an FFT telescope works, we will first summarize traditional aperture synthesis.

We will start by specifying the intensity of all the radio sources in the sky and deriving properties that we can measure. We will then invert the process to recover the intensity. We will use unusual units because it makes the analysis easier; it is straightforward to convert between these units and standard physical units.

We will use separate coordinate systems to describe points in the sky and points near the telescope. Let x, y, and z be coordinates near the telescope. The x axis points east, the y axis points north, and the z axis points up. Everything in the sky is far away, and, because we ignore parallax, all that we care about when we look at a far away point is its direction as seen from the telescope. We will use the wave vector $\vec{k}$ to represent the position of any source in the sky.

If we had magical antennas that did not interact with each other,[1] we could imagine covering everything near the telescope with antennas and recording the voltages forever. The antenna at position (x, y, z) would detect a voltage $A(x, y, z, t)$ at time t. These voltages are just numbers, so we can Fourier transform[2] them to get

$$A(x, y, z, t) = \frac{1}{(2\pi)^2} \int \tilde{A}(k_x, k_y, k_z, \omega) e^{i\omega t - ik_x x - ik_y y - ik_z z} dk_x dk_y dk_z d\omega. \tag{9.1}$$

The functions A and $\tilde{A}$ are samples from a random process. One of the parameters of that process is a map of the radio sources in the sky. These radio sources emit radio waves toward us with some intensity, and antennas on the ground will respond to these radio waves. If we consider radio waves with frequency between ω and $\omega + d\omega$ coming from a small patch of the sky with wave vectors in a volume $dk_x dk_y dk_z$ around $\vec{k}$, then the mean square voltage induced in the antenna would be

$$J\left(\vec{k}, \omega\right) dk_x dk_y dk_z d\omega = \left\langle \left| \tilde{A}(k_x, k_y, k_z, \omega) \right|^2 \right\rangle dk_x dk_y dk_z d\omega$$

[1] Real antennas do interact if they are near each other. This changes their sensitivity as a function of direction. If the antennas are on an evenly-spaced grid, then all of the antennas except the ones near the edge will be affected identically. Otherwise, aperture synthesis algorithms may need to correct for this effect.

[2] The signs are chosen for consistency with the phase of a plane wave in Section 8.3. We use a prefactor of $(2\pi)^{-1/2}$ to make the Fourier transform unitary.

for some generalized function J that is nonzero only on the shell $\omega^2 = c^2k^2$. Here, $\langle\cdot\rangle$ indicates the expected value. The function J parametrizes the random process that generates A and $\tilde{A}$, and we will estimate J. From J, we can calculate the intensity $I_\nu(\vec{k}, \omega)$ by taking into account the details of how the antenna responds to incoming radiation from different directions and the geometrical factors from the curvature of the sky. This is standard and mostly independent of how information is processed in an interferometer.

What we do from here depends on what kind of interferometer we are building.

9.1. Traditional interferometers

A traditional radio interferometer is a bunch of antennas on the ground (i.e. with $z = 0$) pointed in the same direction. The telescope records the voltages A (see equation (9.1)) on each antenna for a short period T of time. It then linearly transforms those voltages into the frequency domain, producing a complex amplitude for each of a number of frequency channels. Each channel represents the component of the voltage at some frequency ω; for the antenna at position (x, y), the transformed output is

$$\begin{aligned} &W(x, y, \omega) \\ =&\frac{1}{2\pi}\int \tilde{A}\left(k_x, k_y, k_z, \omega'\right) e^{-ik_x x - ik_y y} s\left(\omega, \omega'\right) dk_x dk_y dk_z d\omega'. \end{aligned} \tag{9.2}$$

The function $s(\omega, \omega')$ is the combined effect of the Fourier transform relating A to $\tilde{A}$ and whatever algorithm the interferometer uses to transform the measured voltages back into the frequency domain. We assume that s is a reasonable approximation of $\delta(\omega - \omega')$.

For each pair of antennas, the interferometer calculates the quantity

$$g(x_1, y_1, x_2, y_2, \omega) = W(x_1, y_1, \omega)W^*(x_2, y_2, \omega), \tag{9.3}$$

where (x_1, y_1) and (x_2, y_2) are the coordinates of the antennas. For a generic function $\tilde{A}$, this quantity is both complicated and mostly useless. For any value of ω, the values $W(x, y, \omega)$ can we written as a vector $\vec{w}$ with one entry for each of N antennas. Then the values of g are the just the entries of the $N \times N$ matrix $\vec{w}\vec{w}^\dagger$. This matrix has rank one, so its N^2 entries have only N degrees of freedom and just recording the vector $\vec{w}$ seems more economical.

The power of an interferometer comes from an assumption about the random process generating $\tilde{A}$. For any given $\vec{k}$, $\tilde{A}$ encodes the intensity and phase of the signal from that part of the sky. Patches of the sky with different values of $\vec{k}$ are far apart and, for the most part, there is no mechanism to make them coherent with each other. We therefore assume that the phases, and therefore complex values, of $\tilde{A}$ are uncorrelated for different $\vec{k}$, *even conditioned on previous measurements.*

We also assume that, at least over the time it takes to make a measurement, the distribution of $\tilde{A}$ is invariant under time translation. This means that the measured values of W are uncorrelated at different values of ω. If we repeat our measurement of W many times and average each resulting value of g, we therefore obtain the expected value

$$\begin{aligned}
&\langle g\left(x_1, y_1, x_2, y_2, \omega\right)\rangle \\
=&\langle W(x_1, y_1, \omega) W^*(x_2, y_2, \omega)\rangle \\
=&\frac{1}{(2\pi)^2} \int\limits_{\vec{k},\vec{l},\omega',\omega''} \left[\begin{array}{c} \left\langle \tilde{A}\left(\vec{k}, \omega'\right) \tilde{A}^*\left(\vec{\ell}, \omega''\right)\right\rangle e^{-ik_x x_1 - ik_y y_1 + i\ell_x x_2 + i\ell_y y_2} \\ s\left(\omega, \omega'\right) s^*\left(\omega, \omega''\right) d\vec{k} d\vec{\ell} d\omega' d\omega'' \end{array}\right] \\
=&\frac{1}{(2\pi)^2} \int\limits_{\vec{k},\omega',\omega''} \left[\begin{array}{c} \left\langle \tilde{A}\left(\vec{k}, \omega'\right) \tilde{A}^*\left(\vec{k}, \omega''\right)\right\rangle e^{-ik_x (x_1 - x_2) - ik_y (y_1 - y_2)} \\ s\left(\omega, \omega'\right) s^*\left(\omega, \omega''\right) d\vec{k} d\omega' d\omega'' \end{array}\right].
\end{aligned} \tag{9.4}$$

With a long enough measurement period T, the function s can approach a delta function, and the average value of $g\left(x, y, x + \Delta x, y + \Delta y, \omega\right)$ is a good estimate of the function

$$V\left(\Delta x, \Delta y, \omega\right) = \frac{1}{(2\pi)^2} \int \left\langle \left| \tilde{A}\left(\vec{k}, \omega\right)\right|^2 \right\rangle e^{-ik_x \Delta x - ik_y \Delta y} d\vec{k}. \tag{9.5}$$

$$= \frac{1}{(2\pi)^2} \int J\left(\vec{k}, \omega\right) e^{-ik_x \Delta x - ik_y \Delta y} d\vec{k}. \tag{9.6}$$

To obtain a good estimate, at least N measurements must be averaged. Each measurement in the average increases the rank of the matrix of g values by at most one, so N measurements are needed to obtain a full-rank result.

V is called visibility, and it is a function of the distance or baseline $(\Delta x, \Delta y)$ between antennas.[3] An interferometer that samples the visibility densely in the $(\Delta x, \Delta y)$ plane could Fourier transform it to obtain a map of the sky. Most interferometers only sample it sparsely, obtaining one baseline per pair of antennas. These interferometers use more sophisticated techniques to obtain a map [49].

The estimator (9.4) is not biased by independent additive noise in the W values except when measuring the visibility with baseline $\Delta x = \Delta y = 0$. This baseline is called the autocorrelation, and many interferometers ignore it. This results in a loss of information about the average brightness of the sky but not about the variation in intensity in different directions. A careful measurement of the average brightness between 60 and 200 MHz could detect a phenomenon called the global step, but this measurement would likely be made with a very different type of telescope. [50]

[3] Radio astronomers traditionally use coordinates (u, v) in units of wavelength instead of physical distance.

Our assumption about incoherent sources is important and is not necessarily guaranteed to hold. For example, imagine aliens in two different star systems aiming lasers of exactly the same frequency at the Earth. The phase difference between the lasers is random but *constant*. This means that the cross terms in the visibility estimate will not average away, at least within the coherence time of the lasers.

Even if antennas are in a fixed position on the ground, the baseline between them changes as the earth rotates. The visibility data from different times of the day can be combined in a technique called rotation synthesis to sample more of the $(\Delta x, \Delta y)$ plane than would otherwise be possible. If the antennas do not rotate to point in a fixed direction, rotation synthesis algorithms must account for the fact that the antennas are pointing in different directions, and therefore have different spatial responses, at different times of day.

The relationship between the visibility and the intensity distribution in the sky is called the van Cittert-Zernike Theorem, and a more rigorous treatment can be found in [49].

9.2. An aside: polarization

We have, so far, ignored the effects of polarization. Radio waves have two possible polarizations, and an antenna will only record one of them. Some interferometers, including ours, record both and can measure the polarization of the radio signal from the sky. These interferometers use antennas that can collect light from both polarizations and record two signals, A_X and A_Y, one for each polarization. They measure correlations for each baseline:

$$
\begin{aligned}
g_{XX}\left(x_1, y_1, x_2, y_2, \omega\right) &= W_X(x_1, y_1, \omega) W_X^*(x_2, y_2, \omega), \\
g_{XY}\left(x_1, y_1, x_2, y_2, \omega\right) &= W_X(x_1, y_1, \omega) W_Y^*(x_2, y_2, \omega), \\
g_{YX}\left(x_1, y_1, x_2, y_2, \omega\right) &= W_Y(x_1, y_1, \omega) W_X^*(x_2, y_2, \omega), \text{ and} \\
g_{YY}\left(x_1, y_1, x_2, y_2, \omega\right) &= W_Y(x_1, y_1, \omega) W_Y^*(x_2, y_2, \omega).
\end{aligned}
$$

From these, they estimate four visibilities ($V_{XX}, V_{XY}, V_{YX}, V_{YY}$). These visibilities are linearly related to the Stokes I, Q, U, and V parameters. All of these calculations are done the same way as in single-polarization interferometry, even if the Fast Fourier transform technique is used.

The antennas that we use record North-South and East-West linear polarizations and our designs measure the Stokes parameters. We will ignore polarization in the rest of this thesis, except to the extent that measuring the Stokes parameters doubles the number of analog channels needed and quadruples the complexity of the correlator.

9.3. Fast Fourier transform telescope design

As discussed in Section 8.3, if antennas are placed on a grid, then Fourier transforming the signals they measure can produce an image directly. It turns out that this is equivalent to the traditional visibility approach to interferometry.

To estimate the visibility, an interferometer repeatedly computes the frequency-domain signal W (9.2) for each antenna, computes the product

$$W(x_1, y_1, \omega)W^*(x_2, y_2, \omega)$$

for each pair of antennas, and averages. This product estimates the visibility

$$V\left(x_2 - x_1, y_2 - y_1, \omega\right)$$

, and if more than one pair of antennas has the same baseline, then the products of their W functions estimate the same visibility value. An interferometer could, therefore, average the visibility measurements from each pair of antennas that share a baseline.

If the antennas are on a grid, then the average of the g functions (9.3) has a useful structure. For notational convenience, let $W = 0$ at all grid points on which there is no antenna. Then the visibility estimate for each sample of W is

$$g\left(x_1, y_1, x_2, y_2, \omega\right) = W(x_1, y_1, \omega)W^*(x_2, y_2, \omega)$$

and the sum of the visibility estimates of all pairs of antennas with a given baseline is

$$V'\left(\Delta x, \Delta y, \omega\right) = \sum_{x_1, y_1} W(x_1, y_1, \omega)W^*(x + \Delta x, y + \Delta y, \omega), \tag{9.7}$$

which is exactly the spatial autocorrelation function of W. The average of V' divided by the number of antenna pairs that share each baseline is an estimate of V.

By the correlation theorem, V' can be computed by taking the spatial discrete Fourier transform of the W values, computing the square of its magnitude, and Fourier transforming back. The calculation is roughly

$$\sum_{p,q} e^{ip\Delta x + iq\Delta y} \left| \sum_{x,y} e^{ipx+iqy} W(x, y, \omega) \right|^2 \tag{9.8}$$

$$= \sum_{p,q} \sum_{x_1,y_1} \sum_{x_2,y_2} e^{ip(\Delta x + x_1 - x_2) + iq(\Delta y + y_1 - y_2)} W(x_1, y_1, \omega)W^*(x_2, y_2, \omega) \tag{9.9}$$

$$\propto \sum_{x_1,y_1} \sum_{x_2,y_2} \delta_{\Delta x + x_1 - x_2} \delta_{\Delta y + y_1 - y_2} W(x_1, y_1, \omega)W^*(x_2, y_2, \omega) \tag{9.10}$$

$$= \sum_{x_1,y_1} W(x_1, y_1, \omega)W^*(x_1 + \Delta x, y_1 + \Delta y, \omega). \tag{9.11}$$

To make step (9.10) work, p and q must be closely enough spaced to prevent the complex exponential from wrapping around 2π. This can be done with an FFT by padding W with zeros.

This matches the intuition in Section 8.3: the spatial Fourier transform of W is the complex amplitude of the sources in the sky, its square is the intensity distribution in the sky, and the visibility V is the Fourier transform of the intensity distribution in the sky.

The approach to estimating V does not require that the antennas lie on a grid, but if they do then the size of the Fourier transforms is proportional to the number of antennas and the entire calculation runs in time $O(N \log N)$.

This technique is described in [47, 51]. It can be extended to other arrangements of antennas [52].

10. Telescope hardware

In a radio interferometer, antennas receive radio waves from the sky, and the signal from the antennas is converted to a map of the sky in several stages. In this chapter, we will discuss those stages in order. FFT telescopes differ from traditional interferometers only in the algorithms used to analyze data and the physical arrangement of the antennas. It is possible to make the analog hardware for each antenna exactly the same in both designs.

Because we are building an FFT telescope, however, we have different priorities in our analog hardware design. In a traditional interferometer with a large enough number of antennas, estimating the visibilities is the dominant cost. It therefore makes sense to use the best available analog hardware, because better hardware may reduce the number of antennas needed and thus the cost of the digital logic. Better antennas and analog hardware may also reduce the amount of digital processing needed to compensate for any imperfections in the analog components.

The total cost of an FFT telescope is roughly proportional to the number of antennas it uses. Unlike in a traditional interferometer, the cost of analog hardware is likely to be significant even in a very large telescope, so it is important to minimize that cost. We are willing to use less perfect analog components to save on the cost per antenna.

A radio telescope's antennas collect radio waves from the sky. The signal from the antennas is amplified, filtered, possibly shifted in frequency, and recorded by an analog-to-digital converter or ADC. We discuss this process in Section 10.1.

Once the data has been digitized, it is processed, usually in real time, to estimate visibilities as in equation (9.5). In many designs, including ours, this processing is done in three stages. First, an F engine (Section 10.2) converts data from each ADC into a series of samples in the frequency domain, as in equation (9.2). Then a corner turner (Section 10.3) rearranges this data so that, for each channel, the data from every antenna is grouped together. This data is then correlated (Section 10.4) by an X engine or an FFT engine. The output of the correlator is the estimated visibilities. These visibilities are often saved for later software analysis and conversion into maps or pictures of the sky.

Most of an interferometer's real-time digital processing consists of straightforward arithmetic done at high speed. Such computation is often done on field programmable gate arrays (FPGAs), chips that contains a large number of logic gates and simple arithmetic units that can be configured and reconfigured in a few seconds to do different kinds of computations. They are much less expensive and consume much less power than computers or GPUs that have similar performance.

Custom application-specific integrated circuits are even more efficient and, in large quantities, more cost effective, but they are much slower to prototype and can have very large one-time costs.

In this chapter, we discuss each basic real-time component in a radio telescope. We also describe the design constraints of our proof-of-concept and second-generation telescopes and the designs that we chose.

10.1. Amplification and digitization

After a radio signal is received by an antenna, it must be converted to digital form. The final step of this conversion is done by an ADC. An ADC measures a voltage or current many times a second and transmits the result of that measurement in digital form. Each measurement is called a sample, and any given ADC records a certain number of megasamples or gigasamples per second (MSPS or GSPS). Each sample is recorded with a certain number of bits of precision.

ADCs cannot be attached directly to an antenna. They expect an input voltage on the order of one volt, and the voltage that the sky signal induces in the antenna is far less than that. ADCs and the electronics that drive them emit a significant amount of radio-frequency interference, so they should be kept far away or otherwise isolated from the antennas to avoid feeding that noise back into the telescope. Finally, many ADCs do not sample rapidly enough to keep up with the frequencies of interest. The remaining analog components between the antenna and the ADC serve to amplify the signal to an appropriate voltage, transmit it to the ADC, and possibly change the signal's frequency to be a better match for the ADC.

10.1.1. Amplification

All analog devices in the signal chain introduce at least some noise, and most add noise with an effective temperature on the order of room temperature. The average temperature of the sky between 120 and 200 MHz is around 100 Kelvin, depending strongly on frequency. If the signal chain adds noise at 300 Kelvin (around room temperature), then our data will be dominated by analog noise. We avoid this problem by using a low-noise amplifier or LNA as the very first stage of the signal chain. Our LNA amplifies the incoming signal by about 20 dB and has a noise temperature of around 50 Kelvin. The noise from the LNA adds directly to our data, but the effect of additional added noise farther along the signal chain is reduced by a factor of 100. This means that the rest of the signal chain does not need to use complex low-noise components.

After the LNA, the signal is sent through filters and further non-low-noise amplifiers before being digitized.

10.1.2. Digitization and frequency conversion

The final step before digital processing is to convert the analog signal to a useful digital format. There are a few ways to do this using an ADC, and they share some terminology.

All the useful information from the incoming analog signal is within some range around a center frequency. The width of this frequency range is called the bandwidth of the signal, and the signal itself is called the radio frequency or RF signal. The digitized signal is often at a different frequency than the RF frequency; this is called the baseband signal.

We will call the RF center frequency f_{RF} and the sampling frequency f_s. By the sampling theorem, the samples from the ADC can encode a signal losslessly so long as that signal contains no spectral content at frequencies above the Nyquist frequency $\frac{f_s}{2}$. Any frequency above $\frac{f_s}{2}$ will look the same as a different frequency below $\frac{f_s}{2}$; the lower, indistinguishable frequency is called an alias.

10.1.2.1. Direct sampling

The simplest way to digitize an RF signal is to sample the RF signal directly. The RF signal is sent to the ADC through an anti-aliasing filter that removes everything above $\frac{f_s}{2}$. In this case, the RF and baseband signals are identical. The advantage of direct sampling is its simplicity: it needs few components, so there are fewer sources of noise and fewer things to calibrate.

Somewhat higher frequencies can be digitized by taking advantage of aliasing. Frequencies between 0 and $\frac{f_s}{2}$ are called the first Nyquist zone, frequencies between $\frac{f_s}{2}$ and f_s are called the second Nyquist zone, and so on. As long as the RF signal is entirely contained within one Nyquist zone or can be band-pass filtered without losing useful information, the aliases that result from sampling at f_s will all be distinct and the entire signal can be reconstructed digitally. If sampled in an even-numbered Nyquist zone, the frequencies get reversed and must be corrected in software. Sampling in higher Nyquist zones is almost as simple as direct sampling in the first Nyquist zone but is more sensitive to jitter in the ADC. The alignment requirement is also restrictive; for example, if the interesting parts of a signal are between 120 and 200 MHz, then f_s must be between 200 and 240 MHz to use the second Nyquist zone; the third and higher zones are impossible.

10.1.2.2. Heterodyne receivers

A much wider range of input frequencies can be digitized at a given sampling rate by shifting the RF signal's frequency. The general idea is to multiply the RF signal by a sine wave: to shift the frequency by ω_0, one multiplies the signal by $\cos \omega_0 t$. In Fourier space, each spectral component $B \cos(\omega t + \varphi)$ in the original signal becomes $\frac{1}{2} B \cos[(\omega - \omega_0) t + \varphi] + \frac{1}{2} B \cos[(\omega + \omega_0) t + \varphi]$. The output of

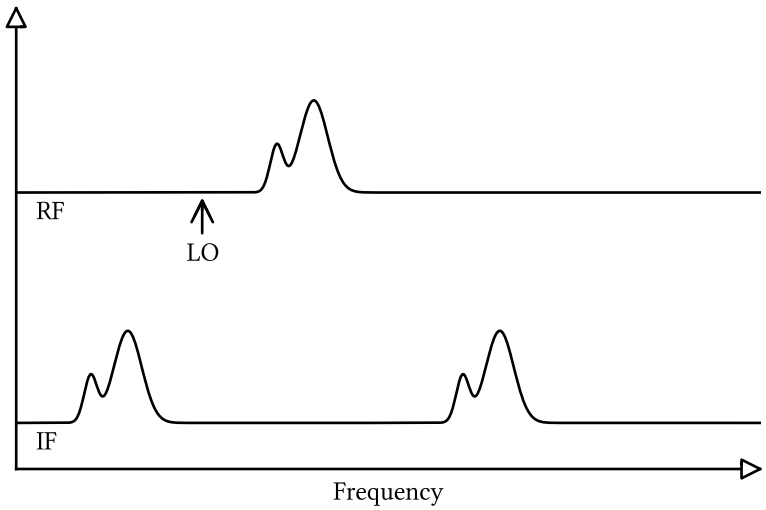


Figure 10.1: Low-side injection downconverts without inverting the spectrum.

the multiplication has two copies of the original signal, one at $\omega + \omega_0$ and one at $|\omega - \omega_0|$. One copy is filtered out and the other is the new intermediate frequency or IF signal. An ADC can digitize the IF signal. The device that generates the sine wave is called a local oscillator to emphasize the fact that it is local to the receiver and is not part of the transmitted signal. This technique is called heterodyning, the device that generates the sine wave at ω_0 is called a local oscillator or LO, and the device that multiplies two analog signals is called a mixer.

To use a heterodyne mixer to downconvert a signal, the local oscillator frequency ω_0 is often set near the frequency of interest. The lower-frequency output of the mixer is kept and the higher-frequency output is filtered out. In low-side injection, ω_0 is below the frequency being received, and the IF output is a copy of the input at a lower frequency (see Figure 10.1). In high-side injection, ω_0 is above the frequency being received, and the IF output has its spectrum reversed as compared to the RF input signal (see Figure 10.2). The local oscillator frequency should not be *in* the RF band of interest; if it is, then the input signal folds over itself and the IF does not contain a useful copy of the RF signal (see Figure 10.3).

One benefit of a heterodyne receiver over direct sampling is that it allows the frequency band that gets digitized to be changed on this fly. This can be done by adjusting the LO frequency and sending the IF signal through a bandpass filter that is narrower than the original RF signal. The LO frequency determines which portion of the RF signal is shifted to the pass band of the filter; those frequencies are preserved and the rest are discarded.

Heterodyne mixers have a significant problem that must be avoided: there are *two* RF input frequencies, called images, that produce each IF output frequencies.

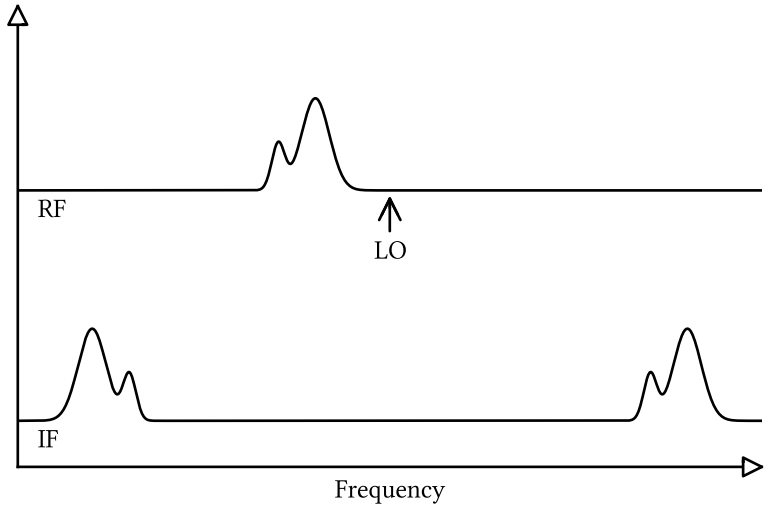


Figure 10.2: High-side injection downconverts and inverts the spectrum.

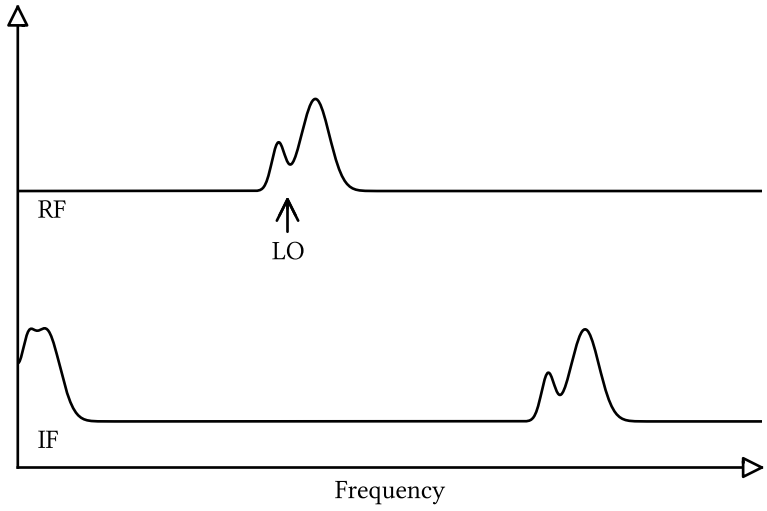


Figure 10.3: If the LO is in the middle of the RF band, the IF signal is folded over.

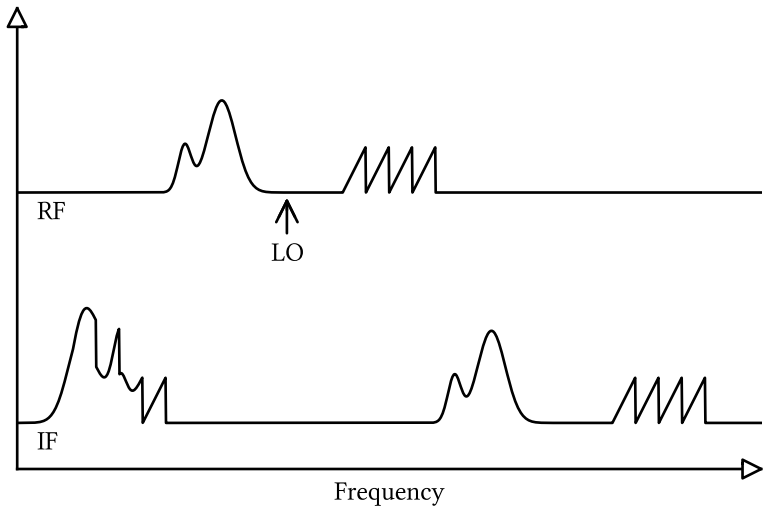


Figure 10.4: Unwanted RF input at the image frequency pollutes the IF output.

Any noise or unwanted signal in the input that is converted to the IF frequency pollutes the output (see Figure 10.4). The unwanted image frequency is the desired RF frequency reflected across ω_0; if ω_0 is near the desired frequency then the image frequency is also near the desired frequency and can be hard to filter out.

Unwanted images are sometimes impossible to avoid with a single heterodyne receiver. Suppose we wanted to design a heterodyne receiver that could tune a 20 MHz RF band between 120 and 200 MHz to an IF of 3 – 23 MHz. To tune the receiver to receive from 140 – 160 MHz, the LO would need to be 137 MHz for low-side injection or 163 MHz for high-side injection. In either case, the image frequency is *in* the band of interest, and any fixed-frequency filter that removes it would also prevent the receiver from tuning the entire band.

This problem can often be worked around by using multiple mixer stages. A receiver could first convert to a IF that is *higher* than the original frequency and use a sharp (and therefore expensive) filter to select a small frequency range. That IF signal is then converted by a second heterodyne stage to a second IF and sampled by an ADC. The filter between the stages eliminates the frequencies that would otherwise be indistinguishable with the desired signal, and the larger frequency shift in the second stage keeps images farther apart and therefore easier to remove. This technique works but is more complicated than we like.

Our proof-of-concept telescope used a heterodyne receiver.

10.1.2.3. Direct conversion or I/Q demodulation

A heterodyne receiver multiplies an RF signal by $\cos(\omega_0 t) = \frac{1}{2}e^{i\omega_0 t} + \frac{1}{2}e^{-i\omega_0 t}$, and its problems with images come from the fact that there are two terms. If we could work with complex-valued signals instead of real-valued signals, then we could multiply by $e^{-i\omega_0 t}$ instead. A spectral component $\cos \omega t$ of the RF signal would turn into $\frac{1}{2}e^{-i(\omega+\omega_0)t} + \frac{1}{2}e^{-i(\omega-\omega_0)t}$. This is a *complex* signal, so it would have no problem with positive and negative frequencies being indistinguishable.

Of course, the signals we work with are voltages on a wire, and voltages are real numbers. But we can simulate complex numbers with two wires: one for the real part and one for the imaginary part. If we take the incoming signal, split it in two, multiply one part by $\cos(-\omega_0 t)$ and the other by $\sin(-\omega_0 t)$, low-pass filter both to remove the image at $-\omega - \omega_0$, and digitize both, then we can reconstruct the full complex signal in software.

Mathematically, if the incoming RF signal is $f(t)$, then the two heterodyned and filtered outputs are

$$I(t) = \text{LPF}\left[\frac{1}{2}\left(e^{-i\omega_0 t} + e^{i\omega_0 t}\right) f(t)\right] \text{ and}$$
$$Q(t) = \text{LPF}\left[\frac{1}{2i}\left(e^{-i\omega_0 t} - e^{i\omega_0 t}\right) f(t)\right].$$

I and Q, individually, are just heterodyned IF signals and they have problems with images. But when we treat I as the real component of a signal and Q as the imaginary component, we have

$$I(t) + iQ(t) = \text{LPF}\left[e^{-i\omega_0 t} f(t)\right].$$

This complex result is exactly the RF input with the frequency shifted down by ω_0. Any real component $\cos(\omega t) = \frac{1}{2}\left(e^{i\omega t} + e^{-i\omega t}\right)$ of the input signal will be appear in the output as $\frac{1}{2}\left(e^{i(\omega-\omega_0)t} + e^{i(-\omega-\omega_0)t}\right)$. If both $\omega - \omega_0$ and $-\omega - \omega_0$ are low enough in frequency to make it through the LPF, then both will be visible in the output, but this is not a problem: there is no other RF image frequency that can be confused with either of these output terms.

This technique is called I/Q demodulation because the RF signal is demodulated into two baseband signals, I and Q or direct conversion because the RF signal is converted directly to baseband. The real or "in-phase" part is called I and the imaginary or "quadrature" part is called Q; hence the name.

Our second-generation telescope takes an RF radio signal from 120 to 200 MHz, shifts it by any amount between 120 to 200 MHz, and low-pass filters the result to select any range of the frequencies it wants without worrying about images.

There are two main downsides to I/Q demodulation: we need a more complicated local oscillator and mixer and two ADCs per channel. This extra complexity makes

calibration more complicated; we discuss this further in Section 12.2. In exchange we get twice the instantaneous bandwidth at any sampling rate, since both the negative and positive IF frequencies are useful.

10.2. F engine

Analog data comes in as a stream of real-valued samples. An F engine converts this data to a sequence of complex amplitudes of different frequency components, as in equation (9.2). The simplest way to do this is to collect a long sequence of samples from the ADC, multiply by a fixed window function, and compute the discrete Fourier transform. If the window is short, this has poor spectral properties – the s function in equation (9.2) will be far from a delta function. We use a polyphase filter bank or PFB to improve the spectral properties of the transformation. A PFB is an efficient way to approach the frequency resolution of a long window using a smaller Fourier transform, and the details are described well in [53].

For some receiver designs, the samples need to be adjusted before or after the FFT. For example, in a direct sampling receiver using an even-numbered Nyquist zone, the frequency channels are reversed. If the receiver uses an I/Q demodulator, then the input to the Fourier transform should be $I(t) + iQ(t)$ instead of I and Q separately. Because PFBs and Fourier transforms are linear, $I(t)$ and $iQ(t)$ can also be independently filtered, Fourier transformed in the complex sense, then added in the frequency domain. The latter approach allows frequency-dependent I/Q calibration to be applied with minimal additional computation (see Section 12.2).

10.3. Corner turn

The F engine performs its calculations independently for each antenna. Any given F engine design can process the data from a certain number of antennas; in a large telescope, there will be many F engines, one for each antenna or group of antennas. The F engines separate the data from each antenna into a large number of much smaller streams of data, one per frequency channel. In the remaining real-time signal processing, data will be processed separately for each frequency channel, but for each frequency channel, the data from each antenna is combined.

The device that makes makes this rearrangement is called a corner turner. The corner turner collects the data from each F engine and outputs the exact same data grouped by frequency for further processing.

Our proof-of-concept telescope had all of its real-time processing on one computer, so the corner turner had nothing to do. Our second-generation telescope uses a 10 gigabit Ethernet switch as its corner turner. This will not scale well to very large devices, and we discuss the problem and a better solution in Chapter 13.

10.4. X or FFT correlator

For each frequency channel, the corner turner outputs the W function equation (9.2) for each antenna. The correlator converts these data to visibilities, as described in Chapter 9. In a traditional interferometer, the device that does this is called an X engine; it outputs visibility data for each pair of antennas. In an FFT telescope, the device that does this is called an FFT engine; it outputs aggregated visibility data for each baseline.

In a simple $m \times n$ FFT telescope, the antennas are on a square grid, so the Δx and Δy values of the baselines cover the range from $(-m, -n)$ to (m, n). This means that there are roughly four times as many baselines as there are antennas. The visibility on the baseline $(\Delta x, \Delta y)$ is the complex conjugate of the visibility on the baseline $(-\Delta x, -\Delta y)$, so one can be ignored and there are roughly twice as many outputs from the FFT engine as there are antennas. In a more complicated hierarchical Omniscope [52], there can be more baselines.

The visibilities computed by the correlator are averaged for a period of time and recorded for further processing to make maps of the sky. Map-making can be a complicated process and is mostly beyond the scope of this thesis. We assume that it will correct for an arbitrary frequency-dependent gain and phase error in the data from each antenna. This correction can be done using closure phase techniques (see, for example, [49]) and can also be done by taking advantage of the inherent redundancy in an FFT telescope [54].

10.5. Proof-of-concept 16-antenna array

We have built two radio telescopes. The first is a proof-of-concept that we designed to build and deploy quickly and inexpensively. It was not a scalable design, but it was meant to teach us about the issues involved in constructing an FFT telescope.

To save time and development costs, we built this telescope entirely from off-the-shelf parts.

Our antennas were simple dual-polarization dipoles. They are designed to function from 120 MHz to 200 MHz and beyond, and each antenna contains an integrated 20 dB low-noise amplifier. The antennas and LNAs are an integrated unit designed for the Murchison Widefield Array (MWA), another Epoch of Reionization instrument being deployed in Australia. The MWA designed these antennas for simplicity and low cost; an antenna with its LNA is available with a short lead time for about $50. The LNAs on these antennas output their signals on 50-ohm coaxial cables, and they are powered by a DC bias voltage on the same cable. We used a Minicircuits bias tee to inject the bias voltage.

We used Universal Software Radio Peripherals (USRPs) from Ettus Research to digitize the signals. A USRP has a USB interface to communicate with a computer and several slots that connect to daughterboards. We attached two TVRX (i.e.

television receiver) daughterboards to each USRP. In this configuration, each USRP can receive two signals from coaxial cables; each signal is downconverted from an adjustable RF frequency to an IF of approximately 20 MHz and then digitized at 64 MSPS. The main bottleneck is the USB interface, which is rather slow. To fit within the USB data rate constraints, the USRP digitally filters the ADC output to a bandwidth of 4 MHz and downconverts to DC before sending to a computer. We connected each USRP to a dedicated Intel D201GLY2A single-board computer. A USRP, two TVRX boards, and the single-board computer cost around $1000 and can receive two channels. This is far less than we would spend to design a customized solution. The receiver design is shown in Figure 10.5.

Each single-board computer was programmed to acquire data from its attached USRP and forward that data to a server for further processing. The server implemented an F engine, corner turner, and FFT correlator in software.

10.5.1. Phase locking and sample synchronization

The USRP has two problems that we needed to work around to make a usable interferometer. The first is that the TVRX receivers are television tuners soldered onto a USRP daughterboard. TV tuners are designed to receive faint signals at variable frequencies, but they are not meant to be used in a coherent antenna array. Each TVRX has its own local oscillator, and there is no reasonable way to synchronize them. As a result, each local oscillator introduces a time-dependent phase error into the data from the attached antenna.

We solved this problem by continuously measuring and correcting the phases. We injected an identical sine wave tone into each TVRX along with the signal from the antennas. We used the tone estimation algorithm in Section 11.2 to measure the phase of that sine wave as seen by each TVRX. We call the raw time-domain data from each TVRX $f_i(t)$ and the measured phase of the sine wave on each TVRX $\varphi_i(t)$. From these, we computed the corrected signal $f_i(t)\, e^{-i\varphi_i(t)}$. As long as each $\varphi_i(t)$ was slowly varying, this gave an excellent corrected signal for each channel.

The second problem was synchronization. The 64 MSPS sampling clocks in each ADC were all derived from the same clock, but USRPs start sampling when they turn on, which means that the first sample on one USRP would not be synchronized with the first sample on another USRP. We worked around this problem by calibrating the time offset on each USRP every time we turned on the array.

We first ran a coarse calibration by broadcasting a timing signal over the network to all of the computers using the IEEE 1588 Precision Time Protocol. By comparing the time at which each computer received a given sample number from its USRP, we could guess the time offset between USRPs to within a few microseconds.

After running the coarse calibration, we injected identical white noise signals into each USRP while running the phase calibration algorithm. This simulated a single point source in the sky directly overhead; therefore, in the absence of a

time offset, the visibilities on all baselines were real. By choosing one channel as a reference and computing the visibility between every other channel and that channel, we could fit the time offset from the visibility phases. This calibration was accurate to around 0.02 samples.

The reference tone and white noise were both generated by an Agilent N9310A signal generator I/Q modulated by an Ettus Research USRP2. The signal was distributed by a Minicircuits zero-phase splitter and injected into each input by a Minicircuits power combiner.

10.5.2. Deploying the proof of concept

We built our proof of concept telescope as a 3×4 array on the roof at MIT. The roof was a convenient location to assemble and debug the instrument, but it was useless for astronomy. Our roof has a direct line of sight to several FM radio transmitters on the roof of Boston's Prudential Tower. The FM radio band is around 100 MHz, which is close enough to our band of interest that our instrument had nowhere near the dynamic range needed to detect any sources in the sky over the interference from the FM radio.

We deployed the array in a 4×4 configuration for a few days at the National Radio Astronomy Observatory in Green Bank, WV. We were able to record signals, but even at the NRAO, interference from local sources was too much for the instrument. The low-noise amplifiers designed for the MWA were meant to be used in Western Australia, where there is essentially no radio interference. In West Virginia, they run in a nonlinear regime and have enough artifacts that it is difficult to recover useful information about faraway radio sources.

10.6. A scalable second-generation telescope

Our second generation telescope is designed with three goals in mind.

- Decreased cost per analog channel. Our proof of concept array used around $500 in parts per analog channel or $1000 per antenna. (Each antenna has two channels, one for each polarization.) This is much higher than we would like in a large scale array. We are willing to spend more money designing components if that effort decreases the final cost of the array.

- Improved analog performance. Our original array could only take data on a 4 MHz band at any given time. It also required complicated workarounds for the phase and timing errors in the USRP receivers, and those workarounds were difficult to work with and introduced noise into our data. For the second generation device, we want higher bandwidth and a clean design that just works.

- Rapid deployment. We built our proof of concept array very quickly, and we are willing to spend longer on the second generation design. Nonetheless, we have a small team, and we want a working device in less time than it would take to design custom silicon or complicated digital devices.

The second generation of our telescope is based on custom hardware. Almost all of the hardware in our array is different from the proof-of-concept. In the proof of concept array, we used off-the-shelf parts. Off-the-shelf parts are useful when they are made in large enough volumes to be inexpensive and are a good match for what we need. The Intel single-board computer is a good example: duplicating even some of its functionality would be impossible for $80 per unit. Small radio components are often on the opposite end of the spectrum. The Minicircuits bias tee, for example, costs more than $50 and contains one capacitor and one inductor. Almost all of the cost comes from the packaging and the fact that it is probably assembled by hand. We therefore try to integrate all necessary functionality onto a small number of circuit boards, and we outsource the assembly of the boards to companies that can make them quickly and inexpensively in almost any volume.

We still use the antennas and low-noise amplifiers from the MWA. They perform well enough, the price is right at $50, and if we need to replace them in the future, we can do so with only minimal changes to the rest of our design.

Near each antenna, we have a small line driver board. The line driver is a custom circuit board that injects 5V to power the LNAs, notch-filters the signal between 88-108 MHz to reduce FM radio interference, amplifies by 57 dB, converts the output to 75 Ω impedance (losing 6–7 dB of power in the process) and outputs the signal for transmission to our receivers. We use 75 Ω RG6 coaxial cable because it performs well and is widely used for cable television, so it is mass-produced, durable, and inexpensive. The line drivers costs about $30 and we use two per antenna, one for each polarization.

Our ADCs and signal processing use hardware from the Center for Astronomy Signal Processing and Electronics Research (CASPER) at Berkeley University. We use two components from CASPER: a board called a ROACH and an ADC board that connects to the ROACH. The ROACH has four 10 gigabit Ethernet ports, a powerful FPGA, and a connector to attach one or two ADC boards. We attach a 64-channel 64 MSPS 12 bit ADC board to each ROACH, although we run the ADCs at 50 MSPS due to clock speed limitations on the FPGA. Each ROACH costs around $5000 (or less if discounted FPGA chips are available), the ADC costs $1600, and the associated cabling costs another $700.

With a maximum sampling frequency of 64 MSPS, directly sampling our signal would put us in at least the fifth Nyquist zone. Making that work well would be difficult at best. The alternative is to use some form of frequency conversion as described in Section 10.1.2.3.

As discussed in Section 10.1.2.2, a single-stage heterodyne receiver is unworkable at our frequencies. A multiple-stage heterodyne receiver would be more complex

and require expensive filters. To avoid those problems, we use I/Q demodulation. We designed a custom receiver board that accepts four channels at 75 Ω. It converts back to 50 Ω (again losing 6–7 dB). It bandpass filters the signal between 120 and 180 MHz (a future design may extend to 200 MHz) and has selectable gain between 9 and 40 dB. It demodulates the signal using an Analog Devices ADL5387 I/Q demodulator, which is available for a few dollars and covers the RF frequency range we need. It low-pass filters the I and Q outputs with a 20 MHz cutoff and sends the signal to the ADC board. I/Q demodulation increases our usable bandwidth from 25 MHz to 50 MHz at the cost of using two ADCs per channel, so the bandwidth of our array and thus overall survey performance per ADC is the same as with a heterodyne receiver. Each receiver costs about $375 per channel.

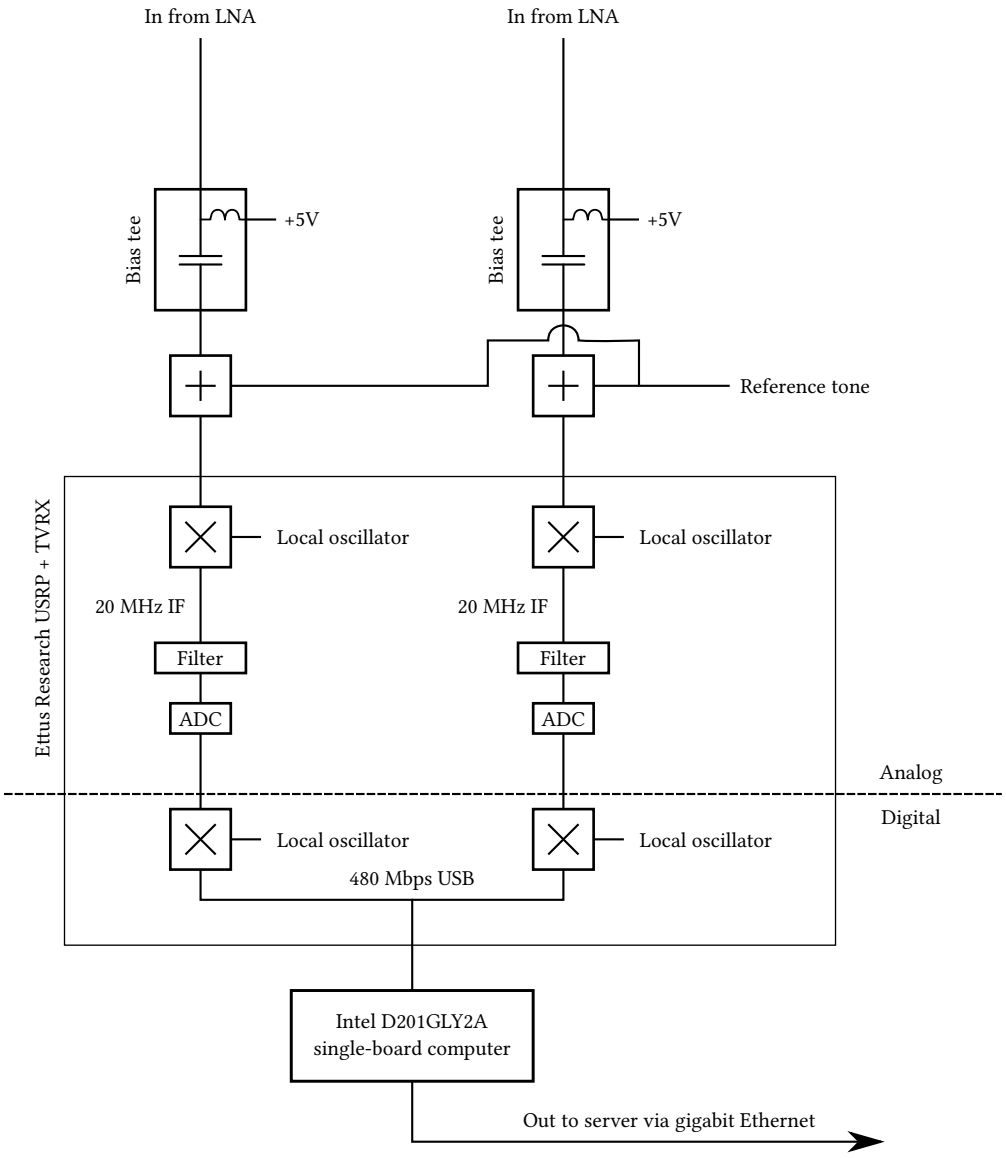


Figure 10.5: Proof of concept receiver for each pair of channels

11. Time-domain capture and its applications

In the process of testing, debugging, and calibrating our hardware, we often need to analyze the data from our ADCs in ways that are tedious to implement on FPGAs. We have developed powerful tools to capture the raw time-domain data from the ADCs and analyze it in software. This allows us to validate the processing done on the FPGAs and to do alternative calculations that can't be done by the F engine.

The simplest applications are to calculate the same things that the real correlator would calculate, but to do them in software. This allows us to compare a high-precision, easy-to-debug software implementation with the FPGA image that we use to take data. This has helped us test many stages of our signal chain and processing [55].

We can also use time-domain capture to develop other tests much more quickly than we could get them working in hardware. For example, we can send specially-crafted signals into the ADCs to measure the nonlinearities in the ADC response.

One of the more interesting applications of time-domain capture is to characterize linear and frequency-shifting analog circuits in response to sine-wave input. The basic idea is generate a sine wave, put it through a splitter, use one output of the splitter as a reference and run the other through the device under test. The difference in the recorded amplitudes and phases of the signals at a given frequency will tell us the device's response. We have used this approach to measure crosstalk and to calibrate our I/Q demodulator. We will discuss the technique we use in Section 11.2.

11.1. Capturing data over a network

As discussed in above, our hardware is optimized to do most of its signal processing on FPGAs. These FPGAs average the data that they collect so that the data that gets sent to a computer is a tiny fraction of the total data collected.

To debug and characterize our system, we want that original data in the time domain, and we need to acquire continuous streams of data that are too large to fit in the FPGA's memory. We do this by selecting a few channels, arranging the data from those channels into packets, and streaming those packets onto 10 gigabit Ethernet. We need to do this in such a way that computer software can keep up with the data.

We format the raw data into UDP packets. Each packet starts with one byte identifying the payload format and a seven-byte sequence number indicating which bytes of the stream it contains. When we start acquiring data, the sequence number

starts at zero, and it gets incremented by the payload size in 8-byte units in each subsequent packet. The remainder of the UDP packet is the payload, which comes in one of three formats.

If we acquire data from a single 8-bit 1 GSPS ADC, then each byte of payload is one sample from the ADC. This gives a data rate of 8 Gbps.

If we acquire data from two 8-bit 1 GSPS ADCs, we decimate each ADC's data by a factor of two on the FPGA. Each two-byte word in the payload is one sample from each ADC. This gives the same data rate of 8 Gbps.

If we acquire data from four 12-bit ADC channels running at 50 MSPS, as used in our telescope, the payload comes in 64-bit blocks. Each block contains one sample from each ADC we select, padded for convenience to 16 bits per sample. This gives a data rate of 3.2 Gbps. We could easily support more than four channels, but four has been enough so far.

All of these data rates ignore overhead from the protocol. Each packet has about 38 bytes of overhead from Ethernet frame headers and gaps, 20 bytes for the IPv4 header, 8 bytes for the UDP header, and 8 bytes for the protocol header, for a total of 74 bytes of overhead. Even at a packet size of 1500 bytes, the overhead will not put us over 10 Gbps.

Each frame carries another kind of overhead, though. We are transmitting to a computer, and the computer needs to do some work for each incoming packet. To minimize the per-frame overhead, we send as large a payload as possible in each packet. Most 10 gigabit Ethernet hardware can handle 9kB packets, but we are limited to about 8 kB due to buffering limits on the FPGA.

On the computer, we have a tool that receives a set number of bytes from the stream for later processing. We do not want to require a top-of-the-line computer to receive data without losing packets, so the program is designed to work as efficiently as possible.

When asked to receive some amount of data, the program allocates enough memory to store all of the captured data and immediately overwrites that memory. That ensures that any page faults needed to make the memory available happen before capturing data. Then the program opens a UDP socket, asks the kernel for a large buffer (to absorb some scheduling latency), and starts receiving packets using the most efficient available mechanism. On Linux, this is the RECVMMSG system call, which allows a program to receive multiple packets of data at once. The program validates the sequence number of each received packet and aborts if any are lost.

Once the acquisition is complete, the program uses SSE4.1 vector instructions to correct the endianness of the data and validate the padding bits. It then saves the raw data to disk or makes it available for our analysis pipeline, in either case taking care to avoid excessive memory allocation or unnecessary page cache pressure. We have basic tools to save the data in raw format, as an ASCII table, or in more specialized formats for processing in MATLAB or Python.

11.2. Tone estimation

When characterizing and debugging our electronics, we often want to create models that describe the response of some part of our system. The behavior of most of our electronics depends, intentionally or otherwise, on the input frequency, so our models express system response as a function of input frequency.

An easy way to measure frequency dependence is to send some input (shaped noise, for example) that covers the frequency band of interest and then to work in the frequency domain. Since our correlator already generates frequency-domain output, we can do a lot of analysis on the output of our correlator in response to different test sources. In practice, however, this can be awkward for several reasons:

- Well-characterized, easily adjustable tone generators are inexpensive; most inexpensive noise sources have very low, fixed output power.
- Some of our electronic devices are nonlinear and we risk confusing different frequencies if we cover our entire bandwidth at once.
- If we inject noise, then we need either careful statistical analysis or very high powers to distinguish the intentionally injected noise from parasitic noise in our circuit.

In many cases, we can do better by sending a single tone into our system, characterizing the response to that tone, and repeating the experiment at different frequencies.

The tones generated by a signal generator are not quite sine waves as seen by another electrical device. The amplitude varies slightly over time, and more importantly, the phase drifts. Even if our tone source were perfect, the clock on our ADCs is not, so *as seen by an ADC*, the phase of any sine wave drifts measurably over even short periods of time (see Figure 11.1). This means that a measured sine wave s might look like

$$\begin{aligned} s(t) &= A(t)\cos(\omega t + \varphi(t)) + n(t) \\ &= \mathrm{Re}\left[\tilde{A}(t)e^{-i\omega t} + n(t)\right] \end{aligned}$$

where ω is the nominal frequency, A and φ are some functions that are slowly varying compared to ωt, $n(t)$ is additive noise, and $\tilde{A}(t) = A(t)\,e^{-\varphi(t)}$ is the complex amplitude of the sine wave. We can usually assume that we know ω exactly, since any small error in ω is equivalent to a small additional time dependence in φ. Our goal is to reconstruct $\tilde{A}$ from s.

To do this, we first find ω. In many cases, ω is already known, but Fourier-transforming s and looking for the maximum value in some range of interest will also give a good estimate.

Recovering the phase of s is slightly tricky: $s(t)$ is a real number and any algorithm that can find its phase will have to recognize its structure on a scale of a few cycles. The process is easiest to understand in terms of complex exponentials. We can rewrite s as

$$s(t) = \frac{A(t)}{2} \exp\left[i\omega t + i\varphi(t)\right] + \frac{A\left(t\right)}{2} \exp\left[-i\omega t - i\varphi(t)\right] + n\left(t\right).$$

We want to separate out the positive and negative frequencies, so we re-scale and frequency-shift the signal to obtain

$$2s(t) \exp\left[-i\omega t\right] \tag{11.1}$$
$$= A(t) \exp\left[i\varphi(t)\right] + A\left(t\right) \exp\left[-2i\omega t - i\varphi(t)\right] + 2n\left(t\right) \exp\left[-i\omega t\right]. \tag{11.2}$$

Then we apply a low-pass filter to eliminate the second term and reduce the effect of the noise. We want this filter to have several properties. It must reject the frequency 2ω strongly enough to eliminate the second term of equation (11.1). This requirement is critical; the tone estimation algorithm does not work at very low frequencies. The filter should be wide enough that it does not destroy the structure of A and φ – any details on a time scale smaller than the inverse bandwidth of the filter will get washed out. This is equivalent to saying that the filter needs to be wide compared to the actual observed bandwidth of the signal so that it does not remove part of the signal. The filter should have a flat phase response and as flat an amplitude response as possible in the range over which the frequency of s will drift. Such filters are relatively easy to construct – any symmetric FIR filter will have a flat phase response, standard windowing algorithms can give reasonably short filters that have an appropriately flat amplitude response, and the overlap-add FIR filtering algorithm is extremely fast. It is critical that the filter attenuates a signal at 2ω strongly so that the second term is removed. Beyond these constraints, the narrower the filter can be, the more of the noise n it will remove. With the equipment we use, a bandwidth of 10 kHz or so is more than wide enough to capture our test signals while keeping unwanted noise under control.

If we normalize the filter so that a DC signal is passed with no change in amplitude and write it as LPF, then the result of the filter is

$$\begin{aligned}\hat{\tilde{A}}\left(t\right) &= \text{LPF}\left[2s(t) \exp\left[-i\omega t\right]\right] \\ &\approx A(t) \exp\left[i\varphi(t)\right] + \text{LPF}\left[2n\left(t\right) \exp\left[-i\omega t\right]\right] \\ &= \tilde{A}(t) + \text{LPF}\left[2n\left(t\right) \exp\left[-i\omega t\right]\right].\end{aligned}$$

The function $\hat{\tilde{A}}\left(t\right)$ is an estimator of the complex amplitude $\tilde{A}$. It magnitude is biased by noise and by any variability in the filter gain in the passband. The former effect is often very small and can be corrected if necessary by measuring the noise,

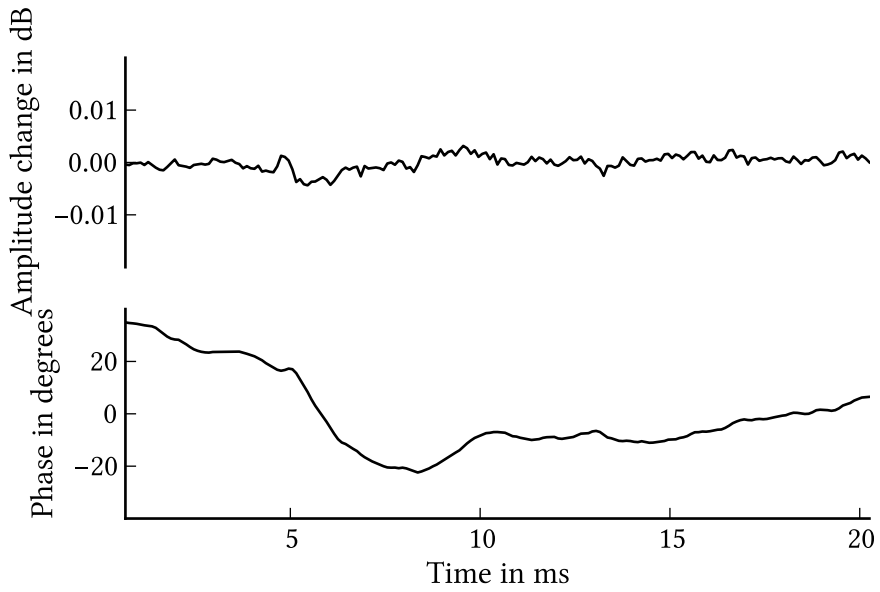


Figure 11.1: The phase of a TTi TGR2050 signal generator drifts measurably relative to our ADC clock over periods of a few ms, while the amplitude is more stable. Both are measured with the tone estimation algorithm.

and the latter effect can be made as small as desired by using a flatter filter. The magnitude $\hat{A}(t) = \left|\hat{\tilde{A}}(t)\right|$ and phase $\hat{\varphi}(t) = \arg \hat{\tilde{A}}(t)$ are estimators of A and φ respectively.

The amplitude estimator can be used directly if the input tone power is known, but the phase is only useful for relative measurements. This is unavoidable because our choice to multiply by $\exp[-i\omega t]$ was arbitrary: there is nothing special about $t = 0$ and we could just as easily have multiplied by $\exp[-i\omega t + \delta]$ for any δ.

12. Real-world effects

There are many real-world effects that change the idealized model of our radio telescope discussed in the previous chapters. This chapter discusses how we measure and correct for some of them.

12.1. Crosstalk

In theory, each antenna and analog signal chain operates independently of the others. We rely on this when we calculate visibilities – aperture synthesis assumes that correlations between antennas result from sources in the sky. If we have sources of noise that are correlated between channels or if signals from one channel leak into another, the resulting correlations will degrade our final image.

Crosstalk, the leakage of signals from one channel to another, is difficult to avoid completely. We need to build a very large number of identical analog signal chains, and for cost- and space-efficiency, we need to pack them close together. All radio-frequency electronics radiate and absorb at least some radiation, and radiation emitted from one signal chain does not have to travel far to affect another signal chain. This couples all of our channels together and introduces correlations in the signal we measure that can, if strong enough, corrupt our image of the sky.

We can mitigate the effects of crosstalk in several ways. We can reduce it by shielding or design changes; we can try to characterize it very precisely and undo its effect digitally, and we can adjust the signal chain to change the crosstalk into a less damaging form. Doing any of these requires us to understand the sources and strength of crosstalk.

12.1.1. Measuring crosstalk

To a very good approximation, crosstalk is a linear time-invariant effect. Cross-talk will couple each frequency between channels independently of any other frequency, and the coupling is characterized by a gain and phase. If we look at the signal at any stage of our signal chain as a vector of complex numbers, one for each channel at a particular frequency, then the crosstalk will be a matrix. More generally, our telescope consists of a number of amplifiers, wires, filters, and so on, all of which are, in principle, identical on each channel. Each of these stages ideally multiplies the signal vector by a scalar, and the effect of crosstalk is to multiply by a matrix instead. All of the scalars commute with all of the crosstalk, so we can think of the cumulative effect of crosstalk as multiplying the ideal signal we would record

by a single matrix. If we call the matrix C, then the recorded signal is $\vec{s}' = C\vec{s}$, where $\vec{s}$ is the ideal (complex) signal we would record in the absence of crosstalk. C is difficult to measure directly because we do not have direct access to the ideal signal $\vec{s}$.

We can measure a related quantity by sending test signals into one channel of our signal chain at a time. Any matrix C with nonzero diagonal elements can be written

$$C = C^{(1)} D,$$

where D is diagonal and all of the diagonal entries of $C^{(1)}$ are 1. If we inject a signal into channel j of our system and put nothing into the other channels, then the signal we would measure in the absence of crosstalk would be zero for all channels other than j. Because D is diagonal, this means that $(D\vec{s})_k = D_{kk} s_k = 0$ for all $k \neq j$. If we measure the actual response $\vec{s}'$ on channels j and k, then we have

$$\begin{aligned} s'_j &= C^{(1)}_{jj} D_{jj} s_j = D_{jj} s_j \\ s'_k &= C^{(1)}_{kj} D_{jj} s_j \end{aligned}$$

and we can divide to obtain

$$C^{(1)}_{kj} = \frac{\vec{s}'_k}{\vec{s}'_j}.$$

If the crosstalk is small, it can be difficult to make this measurement with a noise source: the noise injected into channel j shows up on channel k with its power decreased by a factor of $|C_{kj}|^2$, and if that factor is high enough, then additive noise on channel k will interfere with the measurement. We can minimize this effect by injecting a sine wave $\mathrm{Re}\left[\tilde{A}_j(t)\, e^{i\omega t}\right]$ into channel j and using the technique in Section 11.2 to estimate $\tilde{A}_j(t)$ and the complex amplitude $\tilde{A}_k(t)$ of the signal coupled into channel k. This works well as long as crosstalk occurs on much shorter timescales than the reciprocal of the tone estimator's filter bandwidth.[1] The simplest way to calculate the crosstalk is to averaging $\hat{\tilde{A}}_k \hat{\tilde{A}}_j^{-1}$ over time.

We prefer to calculate the crosstalk somewhat differently. The tone estimator is time-dependent but linear. The estimated input signal $\hat{\tilde{A}}_j$ is the signal from the tone generator plus some noise that makes it through the tone estimator's filter. The signal-to-noise ratio will be excellent because the signal power is high and the filter is narrow. The estimated coupled signal $\hat{\tilde{A}}_k$ is the crosstalk $C^{(1)}_{kj} \tilde{A}_j$ plus noise. The signal-to-noise ratio of $\hat{\tilde{A}}_k$ can be very low; in many cases, the coupled

[1] This is a safe assumption – most of our crosstalk occurs on length scales of a few centimeters. Even if we allow for crosstalk between antennas length scales of a few hundred meters, the maximum time scale is around around 100 µS, which is safe even with a wide filter bandwidth of 10 kHz.

signal can be far smaller than even the small amount of noise that survives the filter. Nevertheless, because $\tilde{A}_j \approx \hat{\tilde{A}}_j$ with extremely small error, we can estimate $C^{(1)}_{kj}$ as the projection

$$\hat{C}^{(1)}_{kl} = \frac{\left\langle \hat{\tilde{A}}^*_j \hat{\tilde{A}}_k \right\rangle}{\left\langle \left| \hat{\tilde{A}}_j \right|^2 \right\rangle}, \tag{12.1}$$

where $\langle \cdot \rangle$ indicates the average over the duration of the experiment. This estimator is sensitive to only one of the n possible noise modes, so the effective noise power is reduced by a factor of the number of samples recorded. We can reject 80 dB of noise with 10^8 samples, which takes two seconds at 50 MSPS. This is the sharpest possible filter and is essentially the same technique that a lock-in amplifier would use to measure the cross-talk at the ADC input. This gives similar results to averaging $\hat{\tilde{A}}_k \hat{\tilde{A}}_j^{-1}$, but it somewhat more efficient and it is easier to understand in terms of noise modes.

We only get the answer at one frequency at a time, but we can automatically sweep the sine wave to get measurements of $C^{(1)}_{kj}$ at any number of frequencies and repeat the process for each pair (j, k), we can measure the entire matrix $C^{(1)}$ one element at a time.

This leaves an unknown degree of freedom D, but this is not a problem. First, unless the circuit has enormous crosstalk, D is probably very close to the identity. Second and more importantly, D is indistinguishable from some gain and phase error that differs for each signal chain, and we already have calibration techniques that can handle that type of error as discussed in Section 10.4.

If we capture more than one channel at a time, we can speed up the computation. The most computationally intensive step is the low-pass filter in the tone estimator, and we can reduce the number of filter evaluations needed. Our crosstalk estimate is

$$\hat{C}^{(1)}_{kl} = \frac{\left\langle \hat{\tilde{A}}^*_j(t) \cdot \text{LPF}\left[2 s'_k(t) e^{-i\omega t}\right] \right\rangle}{\left\langle \left| \hat{\tilde{A}}_j \right|^2 \right\rangle} \tag{12.2}$$

$$= \frac{\left\langle \hat{\tilde{A}}^*_j(t) \cdot \left(\text{LPF}\left[c \hat{\tilde{A}}_j(t) + \text{orthogonal terms}\right]\right) \right\rangle}{\left\langle \left| \hat{\tilde{A}}_j \right|^2 \right\rangle} \text{ for some } c \tag{12.3}$$

$$\approx c \left\langle \hat{\tilde{A}}^*_j(t) \, \text{LPF}\left[\hat{\tilde{A}}_j(t)\right] \right\rangle \left\langle \left| \hat{\tilde{A}}_j \right|^2 \right\rangle^{-1} \tag{12.4}$$

$$= c. \tag{12.5}$$

We can calculate $\hat{\tilde{A}}_j^*(t)$ and $\left\langle \hat{\tilde{A}}_j^*(t)\,\mathrm{LPF}\left[\hat{\tilde{A}}_j(t)\right]\right\rangle \left\langle \left|\hat{\tilde{A}}_j\right|^2\right\rangle^{-1}$ with two filter applications, and we can calculate c for any number of k channels by much less expensive projections.

We can use the ability to measure $C^{(1)}$ to track down sources of crosstalk. If we bypass different parts of our signal chain by injecting the sine wave into different places, we can find the bad parts of the signal chain. The worst source we have found so far turned out to be a cable that connected the ADC chips to the receiver boards.

There are signal processing techniques we can use to reduce the effect of the remaining crosstalk on our data.

12.1.2. Canceling crosstalk

We can correct for crosstalk in real time by multiplying the X or FFT engine input by $\left(C^{(1)}\right)^{-1}$.

If we compute all visibilities with a full X engine, as opposed to aggregating by baseline with an FFT engine, we can also correct for crosstalk when we post-process the the output of the correlator. We assume here that $D = 1$; standard calibration techniques will correct for the diagonal part of the error.

At any particular frequency, suppose that the visibilities in the absence of crosstalk are V_{ij}. The signals before crosstalk are $\vec{s}_i$, so $V = \mathbb{E}\left[\vec{s}\vec{s}^{\dagger}\right]$. With crosstalk, we see

$$\begin{aligned} V' &= \mathbb{E}\left[C^{(1)}\vec{s}\vec{s}^{\dagger}\left(C^{(1)}\right)^{\dagger}\right] \\ &= C^{(1)}\,\mathbb{E}\left[\vec{s}\vec{s}^{\dagger}\right]\left(C^{(1)}\right)^{\dagger} \\ &= C^{(1)} V \left(C^{(1)}\right)^{\dagger} \end{aligned}$$

instead. We can correct the visibility matrix by computing

$$V = \left(C^{(1)}\right)^{-1} V \left(C^{(1)}\right)^{-\dagger}. \tag{12.6}$$

This allows us to measure $C^{(1)}$ and correct it even after taking data. This technique does not work for FFT engine data because we do not have access to the individual elements of V'.

12.1.3. Phase switching

Alternatively, we can use a technique called phase switching in the analog signal chain to reduce the impact of crosstalk on the estimated visibilities. The idea behind phase switching is to scramble the analog signal as early as possible and undo the

scrambling digitally. If most of the crosstalk happens on the scrambled part of the signal, then the crosstalk will look different from a real sky signal.

The simplest kind of scrambling to use is to physically multiply each analog channel i by a time-dependent real number $q_i'(t)$. After digitizing the channel, the digital logic will divide by $q_i(t)$, canceling the effect. (Ideally $q' = q$, but the multiplication is done by an analog device and there will be some error.) To simplify the analysis, we assume that each $q(t)$ and $q'(t)$ is constant over each window transformed by the F engine. If all of the crosstalk happens after multiplying the signal by q', then the correlation between channels i and j in the frequency domain after dividing by q is

$$\begin{aligned} V_{ij}' &= \left\langle q_i^{-1}\left(q_i' s_i + q_j' C_{ij}^{(1)} s_j\right)^* q_j^{-1}\left(q_j' s_j + q_i' C_{ji}^{(1)} s_i\right)\right\rangle \\ &= \left\langle \frac{q_i' q_j'}{q_i q_j} s_i^* s_j + \frac{q_i'^2}{q_i q_j} C_{ji}^{(1)} |s_i|^2 + \frac{q_j'^2}{q_i q_j} C_{ij}^{(1)} |s_j|^2 + \frac{q_i' q_j'}{q_i q_j}\left(C_{ij}^{(1)}\right)^* C_{ji}^{(1)} s_j^* s_i \right\rangle. \end{aligned}$$

If the q functions are chosen so that

$$\left\langle \frac{q_i'^2}{q_i q_j} \right\rangle = \left\langle \frac{q_j'^2}{q_i q_j} \right\rangle = 0, \tag{12.7}$$

then the first-order terms average to zero. If the analog multipliers are perfect, then $q = q'$. In this case, there is a simple way to satisfy equation (12.7): make each q_i an independently random sequence of $+1$ and -1, changing every few F engine windows. Switching the multipliers between $+1$ and -1 is called phase switching. Each F engine cycle makes $\frac{q_i'^2}{q_i q_j}$ equal $+1$ or -1 with equal probability, so the first-order effect of crosstalk on visibility falls as $N^{-1/2}$ where N is the number of phase switches averaged.

The cancellation can be sped up by making each q_i periodic and selecting the periodic sequences for each i so that they are all mutually orthogonal. This way the cancellation is exact after any integral number of periods. The q_i sequences can be Walsh codes, for example.

Both of these constructions are robust against imperfect analog multiplication. Suppose that $q_i' = a q_i + b$ where $a \approx 1$ and $b \approx 0$. Then the first-order coefficients have the form

$$\frac{q_i'^2}{q_i q_j} = \frac{(a q_i + b)^2}{q_i} q_j^{-1}.$$

In the random case, q_j^{-1} is independent of the other factor and $\left\langle q_j^{-1} \right\rangle = 0$ so the first-order error still averages away. In the case where q_i and q_j are orthogonal sequences, let $\vec{c}$ be the vector of values that q_i cycles through and $\vec{d}$ be the vector of values that q_j cycles through. Let n be the length of the vectors. Then after a

single cycle,

$$\left\langle \frac{q_i'^2}{q_i q_j} \right\rangle = \frac{1}{n} \left[\left(a^2 + b^2\right) \vec{c} + 2ab \right] \cdot \vec{d}.$$

This is identically zero if $\vec{c}$ and $\vec{d}$ are orthogonal and the average value of $\vec{d}$ is zero. Therefore, using orthogonal periodic phase switching sequences works even if the multipliers behave differently for each channel and for $+1$ and -1 as long as all of the sequences are orthogonal to the sequence of all ones. A complete set of Walsh codes should not be used: the all ones vector should be omitted.

Phase switching was invented by Martin Ryle, and our phase switching design is described in detail in [53]. It works equally well with an X or FFT engine.

12.2. I/Q imbalance and quadrature phase errors

Our telescope design uses I/Q demodulation as described in Section 10.1.2.3 to downconvert the RF signal before digitization.

The I/Q demodulator we use takes a single local oscillator signal at $2\omega_0$, divides it into two signals at ω_0 90° apart, and multiplies by them. Then it passes both signals through similar low-pass filters. This is different from an ideal I/Q demodulator in two main ways.

- *Quadrature phase error.* The local oscillator-derived carriers $\cos(-\omega_0 t)$ and $\sin(-\omega_0 t)$ may not differ by exactly 90 degrees. In practice, they seem to differ by a stable phase for any particular ω_0.
- *Mismatched filters.* The two low-pass filters might be different. In particular, they might have different frequency-dependent gains and phases. (We can lump differing amplitudes of the carriers into this error as well.)

We can analyze and measure these errors in terms of a few parameters. The quadrature phase error is a single angle φ such that, instead of multiplying by $\sin(-\omega_0 t)$, the second mixer multiplies by $\sin(-\omega_0 t + \varphi)$. The filter mismatch can be modeled as a second filter applied to Q with gain response $g(\omega)$ and phase response $\theta(\omega)$. Since Q is a *real* signal, $g(\omega)$ must be even and $\theta(\omega)$ must be odd.

This parametrization is useful for understanding an I/Q demodulator and validating our calibration. In our telescope, we use a more general model that accounts for other types of linear errors that violate these assumptions. We can correct these errors in real-time or, with an X engine, after taking data.

12.2.1. Quadrature phase and filter mismatch parameters

The net effect of these errors is not a linear time-independent function on the complex signal $I(t) + iQ(t)$: it mixes non-trivially between positive and negative frequencies. It can still be treated analytically in the frequency domain, but, in our

experience, this results in an awkward and inefficient method for measuring φ, g, and θ.

Instead, we work in the time domain and consider the effect of these errors on inputs consisting of a single tone. Suppose that we send a signal

$$B\cos\left((\omega+\omega_0)t+\delta\left(t\right)\right)$$

into our receiver. We do not phase-lock our signal generator to the ADC, so $\delta\left(t\right)$ is some slowly-varying phase error. In general, the amplitude B of the signal is also unknown, but we will not need the overall amplitude.

Suppose that the LPF has gain h_{LPF} and phase ε_{LPF} at the frequency ω. Our model pushes all of the errors into the Q terms, so the I output will be

$$\begin{aligned}
I\left(t\right) &= \text{LPF}\left[\frac{A}{4}\left(e^{-i\omega_0 t}+e^{i\omega_0 t}\right)\left(e^{i(\omega+\omega_0)t+\delta(t)}+e^{-i(\omega+\omega_0)t-\delta(t)}\right)\right] \\
&\approx Ah_{\text{LPF}}\cdot\frac{1}{4}\left(e^{i(\omega t+\delta(t)+\varepsilon_{\text{LPF}})}+e^{-i(\omega t+\delta(t)-\varepsilon_{\text{LPF}})}\right) \\
&= Ah_{\text{LPF}}\cdot\frac{1}{2}\cos\left(\omega t+\delta\left(t\right)+\varepsilon_{\text{LPF}}\right) \\
&= Ah_{\text{LPF}}\cdot\frac{1}{2}\cos\left(\left|\omega\right|t+\left[\delta\left(t\right)+\varepsilon_{\text{LPF}}\right]\text{sgn}(\omega)\right).
\end{aligned}$$

We assume that the LPF's response $h_{\text{LPF}}e^{i\varepsilon_{\text{LPF}}}$ is independent of δ. The other terms will be so far from the passband as to be negligible. On the other hand, Q will be affected by all of the errors. The observed output will be

$$\begin{aligned}
&Q^{(\text{meas})}\left(t\right) \\
=&\,\text{ERR}\circ\text{LPF}\left[\frac{A}{4i}\left(e^{-i\omega_0 t+\varphi}-e^{i\omega_0 t-\varphi}\right)\left(e^{i(\omega+\omega_0)t+\delta(t)}+e^{-i(\omega+\omega_0)t-\delta(t)}\right)\right] \\
=&A\cdot\text{ERR}\left[h_{\text{LPF}}\cdot\frac{1}{4i}\left(-e^{i(\omega t+\delta(t)+\varphi+\varepsilon_{\text{LPF}})}+e^{-i(\omega t+\delta(t)+\varphi-\varepsilon_{\text{LPF}})}\right)\right] \\
=&Ah_{\text{LPF}}g\left(\omega\right)\left[\frac{1}{2}\sin\left(\omega t+\delta\left(t\right)+\varphi+\theta(\omega)+\varepsilon_{\text{LPF}}\right)\right] \\
=&Ah_{\text{LPF}}g\left(\omega\right)\left[\frac{1}{2}\cos\left(\omega t+\delta\left(t\right)+\varphi+\theta(\omega)+\varepsilon_{\text{LPF}}-\pi/2\right)\right] \\
=&Ah_{\text{LPF}}g\left(\omega\right)\left[\frac{1}{2}\cos\left(\left|\omega\right|t+\left[\delta\left(t\right)+\varphi+\theta\left(\omega\right)+\varepsilon_{\text{LPF}}-\pi/2\right]\text{sgn}(\omega)\right)\right].
\end{aligned}$$

We are again assuming that δ has no effect on the filter mismatch error ERR.

Using time-domain capture as described in Section 11.2, we can estimate the time-dependent complex amplitudes $\tilde{A}_I$ and $\tilde{A}_Q$ of the cosines wave at frequency

$|\omega|$ recorded in I and $Q^{(\text{meas})}$. The magnitudes are

$$\left|\tilde{A}_I(t)\right| = \frac{A h_{\text{LPF}}}{2} \text{ and}$$
$$\left|\tilde{A}_Q(t)\right| = \frac{A h_{\text{LPF}}}{2} g(\omega);$$

their ratio is $g(\omega)$. The phases are (up to an unknown but identical constant)

$$\arg \tilde{A}_I(t) = \delta(t) + \varepsilon_{\text{LPF}} \operatorname{sgn}(\omega) \text{ and}$$
$$\arg \tilde{A}_Q(t) = \delta(t) + (\varphi + \varepsilon_{\text{LPF}} + \theta(\omega) - \pi/2) \operatorname{sgn}(\omega);$$

their difference is $(\varphi + \theta(\omega) - \pi/2) \operatorname{sgn}(\omega)$. To correct for the sign flip and $\pi/2$ term, we can compute

$$B = \begin{cases} i \left\langle \hat{\tilde{A}}_I^{-1}(t) \hat{\tilde{A}}_Q(t) \right\rangle & \text{if } \omega > 0 \\ i \left\langle \hat{\tilde{A}}_I^{-1}(t) \hat{\tilde{A}}_Q(t) \right\rangle^* & \text{if } \omega < 0 \end{cases}$$

where $\langle \cdot \rangle$ is the time average. Then $|B|$ is an estimator for $g(\omega)$ and $\arg B$ is an estimator for $\varphi + \theta(\omega)$. By repeating for different values of ω, we can obtain accurate estimates of $g(\omega)$ and $\varphi + \theta(\omega)$ as functions of ω. Because φ is constant and θ is odd, we can fit them separately from $\arg B$.

12.2.2. Generalized I/Q calibration

We can unify all three errors g, θ, and φ as matrices in the frequency domain. For any $\omega > 0$, if we inject a signal with complex amplitude $\tilde{A}_+$ at frequency $+\omega$ and a signal with complex amplitude $\tilde{A}_-$ at frequency $-\omega$, then the ideal response of the I/Q demodulator is

$$\begin{bmatrix} \tilde{I}^{(\text{ideal})} \\ \tilde{Q}^{(\text{ideal})} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -i & i \end{bmatrix} \begin{bmatrix} \tilde{A}_+ \\ \tilde{A}_- \end{bmatrix}.$$

The effect of the g, θ, and φ errors is to change the response to

$$\begin{bmatrix} \tilde{I} \\ \tilde{Q} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -ige^{i\varphi + i\theta} & ige^{i\varphi - i\theta} \end{bmatrix} \begin{bmatrix} \tilde{A}_+ \\ \tilde{A}_- \end{bmatrix} \tag{12.8}$$
$$= E \begin{bmatrix} 1 & 1 \\ -i & i \end{bmatrix} \begin{bmatrix} \tilde{A}_+ \\ \tilde{A}_- \end{bmatrix} \tag{12.9}$$

where $E(\omega > 0)$ is a 2×2 matrix characterizing the I/Q demodulator error. This means that all three errors just multiply the pair I and Q by a matrix E. As long as φ is small, E is far from singular.

We can measure E in a more general model. Injecting a tone at $+\omega$ is equivalent to setting $\tilde{A}_+ = a_+ e^{i\xi_+}$ and $\tilde{A}_- = 0$ and injecting a tone at $-\omega$ is equivalent to

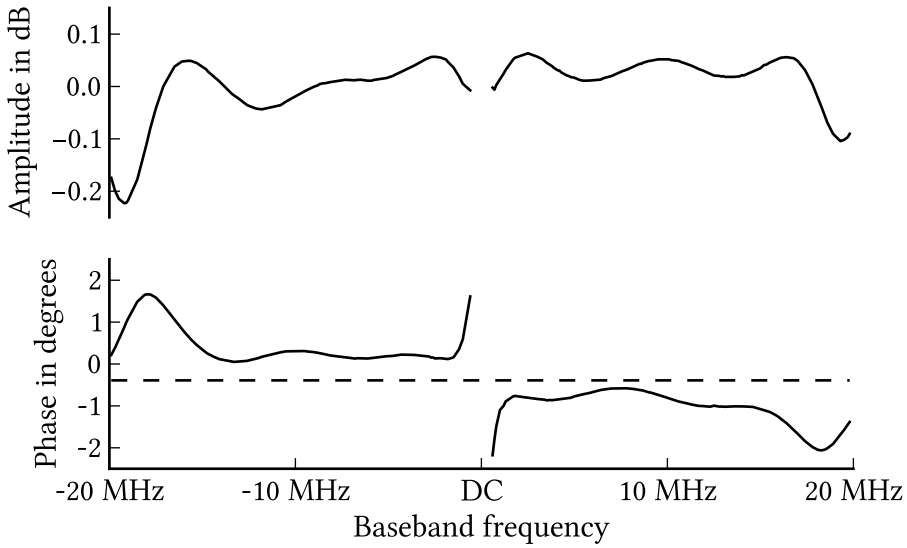


Figure 12.1: One of our receiver's I/Q calibration data nearly fits our model. This is the amplitude and phase of B. The quadrature phase error is the dashed line; it is close to 0.5 degrees as specified by Analog Devices. The apparent g and θ are not quite odd; the deviation may be due to the two multipliers in the ADL5387 demodulator having slightly different frequency responses. Data near DC is omitted because our ADCs cannot handle very low frequencies. The points on this plot were measured in random order and were not smoothed in any way; tone estimation produces very clean results.

setting $\tilde{A}_- = a_- e^{i\xi_-}$ and $\tilde{A}_- = 0$ for some amplitudes $a_\pm$ and phases $\xi_\pm$. If the tone generator is well-calibrated, then $a_\pm$ are known. Because the phases drift over time, $a_\pm$ and $\xi_\pm$ are slowly-varying functions of time.

We have no concept of overall phase, so $\xi_\pm$ are unknown. We can assume that, in any measurement taken with the I/Q demodulator, the phase of the I channel is correct. We therefore remove two degrees of freedom by forcing E_{11} and E_{12} to be positive real numbers. One of these degrees of freedom is unnecessary: it just affects the global phase offset of the I/Q demodulator. The other allows positive and negative frequencies to have different relative phases; this will not affect our telescope at all because, once I/Q demodulation is done, we treat positive and negative frequencies completely independently.

Under these constraints, we can solve for E directly by using tone estimation to measure the (time-dependent) $\tilde{I}$ and $\tilde{Q}$ responses to the positive- and negative-frequency input signals. The measured magnitudes of $\tilde{I}$ tell us the top row of E, the measured phases of $\tilde{I}$ tell us $\xi_\pm$, and we can read off the bottom row of E from the measured values of $\tilde{Q}$. As before, we do this using tone estimation and averaging the measurements over time.

This generalization is equivalent to allowing g and θ to be general functions instead of odd functions and adding an additional parameter to characterize the frequency-dependent gain error on I. This captures mismatched frequency responses in the inputs of the I and Q mixers as well as the matching part of the output filter responses.

12.2.3. Correcting I/Q errors

If we know the matrix $E(\omega)$ in advance, either by measuring it directly or by solving equations (12.8) and (12.9), then we can correct it in the F engine. We convert I and Q to the frequency domain independently and keep only the positive frequencies of the result. Then, for each frequency ω, we multiply the result by

$$\left(E(w) \begin{bmatrix} 1 & 1 \\ -i & i \end{bmatrix} \right)^{-1}$$

to recover the positive and negative frequency components.

If we treat the I and Q channels as separate antennas, then this correction has exactly the same form as crosstalk between channels, except that the diagonal part is not close to 1. If we use an X engine, then we can correct it after taking data using the technique in Section 12.1.2.

13. The Butterfly Network

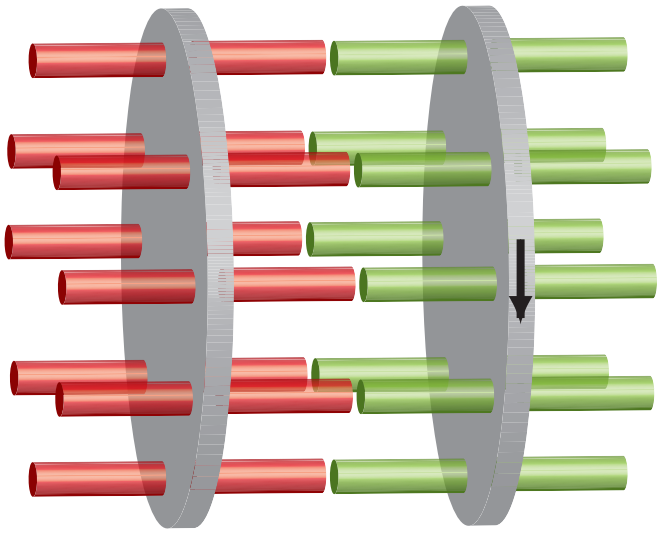


Figure 13.1: Toy example of how the $N = 8$ corner turning problem could be solved using moving parts, with 8 devices simultaneously transmitting data into the 8 red/dark grey links on the left and 8 other devices receiving data from the 8 green/light grey links on the right. After 8 successive $45°$ rotations of the right wheel, all input devices have transmitted the required data to all output devices.

So far, we have discussed two significant costs of a radio telescope: the cost per antenna of the analog hardware and the cost of the required computation as a function of the size of the array. Using off-the-shelf parts, we can get the cost per antenna low enough for a large array, and with the FFT correlation engine the computation cost can be made to scale as $O(N \log N)$.

We have ignored a third cost that becomes important in very large instruments, though: the cost of moving data. The difficult part is what is known as the *corner turning problem*. Each ADC and F engine produces a stream of data, and all of that data needs to be rerouted to the X, FFT, or MOFF engine that processes data at each frequency. The problem is essentially that of rapidly transposing a matrix that is too large to store on one single device: each correlator engine needs to see some frequency slice of the data from each antenna in the array. Suppose

that the output of each F engine is a stream of samples from each of M separate frequency channels.[1] If we imagine all this data arranged in an $N \times M$ matrix, then each *row* of the matrix comes from one F engine. However, the subsequent computation of UV plane visibilities needs to combine information from all N antennas, separately for each frequency, i.e. each *column* of the matrix needs to be processed separately. Transposing this matrix is called corner turning. Regardless of the correlator design used, the amount of computation of required for the second stage is at least proportional to N, so we need $M \gtrsim N$ channels to distribute the computation. In either case, the corner turning is a major bottleneck, since transferring data between all $N \times M$ pairs of first-stage and second-stage devices requires moving the entire contents of the matrix across a large network. For example, for an $N = 64 \times 64$ dual polarization array sampling at 400 MHz, the corner turner has to route about 13 terabytes per second. Once this bottleneck has been passed and the second stage has been completed, however, the resulting sky maps (or their Fourier transforms) can be time-averaged, dramatically reducing the data rate to manageable levels.

Modern radio telescopes have typically adopted one of the following solutions to the corner turning problem:

1. Writing the entire matrix to a single, extremely fast, giant memory module where it can be read out transposed, or using some other device with size $O(M^2)$, for example enough wires to turn the corner directly.

2. Routing all the data through an off-the-shelf non-blocking switch.

3. Using enough wires to make all the connections directly.

The first approach has been used by numerous experiments, and the second has been successfully implemented in the packetized CASPER correlator [56, 57] used by the PAPER experiment [58], where $N = 32$ (including polarization) is small enough to be handled by a single 10 GB Ethernet switch. The third is used in some very large correlators such as EVLA and ALMA. Unfortunately, all of these approaches become expensive for very large N, which makes it timely to explore alternative solutions.

Another way of thinking about the corner turning problem is that it involves the only part of an interferometer that is not embarrassingly parallel[2]: it is easy to build many antennas, many A/D converters, many time-domain Fourier transformers and many correlators acting on separate frequency bands. The corner turn is the piece that transposes the data matrix to keep the processing embarrassingly parallel.

[1] For simplicity, we ignore the polarization issue in our discussion, since it can be trivially incorporated by simply doubling N and treating each of the two polarization channels from each antenna as an independent data stream.

[2] Computer scientists say that a problem is "embarrassingly parallel" if it can be trivially distributed across a large number of processors that do not need to communicate with each other.

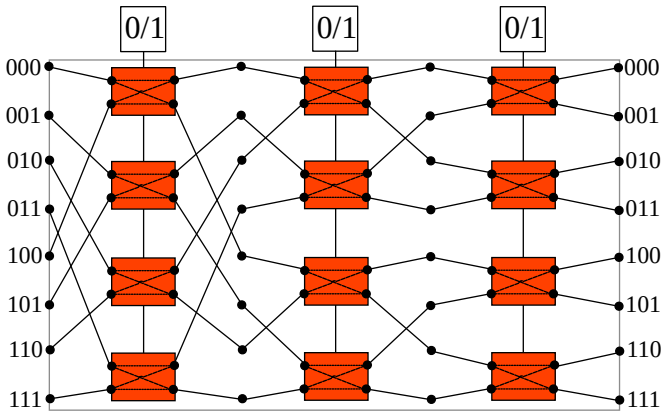


Figure 13.2: How to solve the $N = 8$ corner turning problem using a butterfly network, with 8 devices simultaneously transmitting data from the left and 8 other devices receiving data from the right. After the three control bits shown at the top have looped through all 8 combinations 000, 001, 010, 011, 100, 101, 110 and 111, all input devices have transmitted the required data to all output devices. The boxes ("controlled swappers") have two input wires and either pass them straight through to their output wires or swap them, depending on whether the control bit (drawn as entering from above) is 0 or 1, respectively. The 8 inputs are numbered in binary on the left hand side, and we see that the 1^{st} row of swappers can flip their 1^{st} address bit, the 2^{nd} row of swappers can flip their 2^{nd} bit, and the 3^{rd} row of swappers can flip their 3^{rd} bit.

The rest of this paper is organized as follows. In Section 13.1, we present our solution to the corner turning problem, which requires neither general-purpose network switches nor additional memory. We discuss various physical implementation options in Section 13.2.

13.1. The butterfly algorithm

Below we will limit our discussion to the special case where $M = N$, i.e., where the matrix to be transposed is square, or, equivalently, where there are equal numbers of devices writing to and reading from the corner turner, since if one has an efficient corner turner for this case, then the general case becomes easy to solve. For example, one can simply pad the matrix with zeros, which is equivalent to inserting dummy data sources or sinks, or combine a few adjacent entries into one larger entry, splitting it at the end if necessary [59]. We will present an optimized solution for the case where N is a power of 2.

13.1.1. The problem

In the discussion below, we will refer to the concept of a "link," by which we mean some connection between two computers, FPGAs, or any other devices (nodes) that can carry data at a rate of N matrix entries each time a matrix is transposed. In most cases, this will simply be the rate at which the f stage of the correlator outputs data *for a single antenna.* In other words, the data rate on a given link is independent of the size of the interferometer. If the data rate for a single channel exceeds that of the technology we use to build our corner turner (e.g. 1 Gbps if we used gigabit Ethernet connections), then we will use bundles of identical connections as our links.

It is easy to solve the corner turning problem with a non-blocking switch that can connect $2N$ links: each source node that starts with one row of the matrix simply transmits all of its data, with each entry in the matrix addressed to the sink node labeled with the column number of that entry. Each node sends or receives at exactly the link rate, so a general-purpose nonblocking switch can handle all of the data.

Large general-purpose high-speed switches are expensive because they are fully non-blocking, allowing any set of input devices to simultaneously transmit to any set of output devices. This is overkill for the corner turning problem, since we have complete prior knowledge of how data needs to be distributed. This suggests the possibility of reducing cost by giving up complete generality.

We need each of the N source nodes to transmit data through our corner turner to each of the N sink nodes, with each source node transmitting exactly a $1/N$ fraction of its total data rate to each sink node. This can be done with a very restricted kind of switch using no memory at all. Such a switch has N states, selected by a control input $c \in \{0, \ldots, N-1\}$, where the source node labeled with a number i is connected to the sink node $j = p(c, i)$ where the function p has the following properties:

1. For fixed i, all $p(c, i)$ are unique and in the range $0, ..., N-1$.
2. For fixed c, all $p(c, i)$ are unique and in the range $0, ..., N-1$.

In other words, the corner turner performs a different permutation of the inputs at each time step, such that after N steps, every input node has been connected to every output node exactly once.

With such a switch, each source node i transmits the i, j entry of the matrix exactly when $p(c, i) = j$, and each sink node j will receive the entire column j, albeit in some arbitrary order. This means that each source node transmits data in a different order, but most receiver or f stage designs should be able to handle this without difficulty.

13.1.2. Our solution

How should we choose the sequence of permutations p? There are clearly vast numbers of permutation sequences that satisfy the two requirements above, since we have $N!$ choices even for $p(0, j)$ alone.

13.1.2.1. A mechanical solution

One simple solution is that defined by the cyclic permutations

$$p(c, i) \equiv c + i \bmod N.$$

This choice is illustrated in Figure 13.1 for a toy example where where $N = 8$. If we connect the input devices to the metal bars protruding on the left side and the output devices to the bars protruding to the right, then the N successive 45° rotations of the right wheel will achieve a complete corner turn where every input device has transmitted to every output device.

13.1.2.2. The butterfly algorithm

In practice, one of course needs to accomplish all operations electronically without large moving parts. An elegant method for implementing precisely the cyclic permutations of Figure 13.1 electronically was discovered about a decade ago by Lynn Urry and implemented for the Allen Telescope Array [60, 61], but this was unfortunately never published in a journal and did not become as widely known as it deserves to be. The other authors of this paper independently discovered the methods that we will describe below, which have the further advantage of being even cheaper to implement.

We schematically illustrate a simple solution in Figure 13.2, where the boxes ("controlled swappers") have two input wires and either pass them straight through to their output wires or swap them, depending on whether a control bit (drawn as entering from above) is 0 or 1, respectively. If the $N = 8$ inputs i are numbered in binary, then the 1^{st} row of swappers can flip their 1^{st} bit, the 2^{nd} row of swappers can flip their 2^{nd} bit, and the 3^{rd} row of swappers can flip their 3^{rd} bit. This means that this corner turner implements the permutations

$$p(c, i) \equiv c \operatorname{xor} i,$$

where the integers c, i and j are written in binary on the top, left and right sides of Figure 13.2, respectively.

This basic network topology where a given node successively "talks" with nodes separated by 2^0, 2^1, 2^2, etc. appears in a wide range of electrical engineering and software applications, including the Fast Fourier transform of N numbers, and is often referred to as a "butterfly network". When the "talk" part is a swapper like in Figure 13.2, the resulting network is a special case of a Banyan network

[62] – a type of general-purpose network switch which is nonblocking for certain permutations (as opposed to fully nonblocking, which would allow nodes to talk to each other in any permutation.) A key point about the corner turner that we are proposing is that we are *not* using it as a general-purpose switch but rather with a specific algorithm: to cycle through N very particular permutations, which is precisely what is needed to solve the problem at hand.

For comparison, the method in [60, 61] also uses a Butterfly network, but changes all $N \log N$ controlled swappers independently at each control step c instead of using the same setting for each of the $\log N$ columns. The latter implementation thus requires N times fewer control input wires, and we will see below how it can be further simplified to cut cost.

The butterfly algorithm we have proposed requires that N be a power of 2. As we will see in 13.2 , the cost of a butterfly corner turner is likely to constitute only a small fraction of the total cost of a large N radio array, so for a general number of antennas, a one can simply round N up to the nearest power of two for the corner turner.

13.1.2.3. An even cheaper corner turner using perfect shufflers

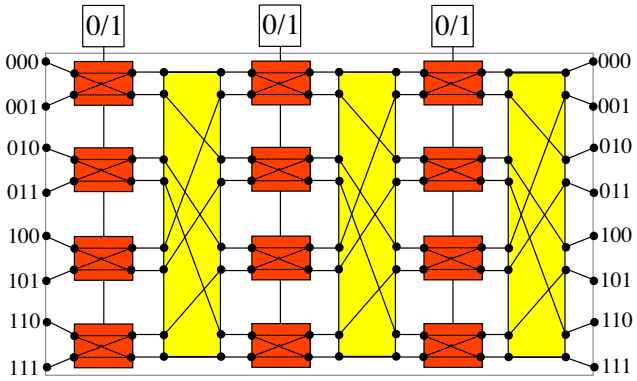


Figure 13.3: An equivalent network to Figure 13.2, but reorganized so that the connections after each row of swappers are identical. These connections, contained within the tall boxes, are seen to correspond to a "perfect shuffle", whereby the top half gets interleaved with the bottom half, and corresponds to cyclically shifting the address bits that label the inputs on the left-hand-side.

An obvious drawback to the configuration in Figure 13.2 is that the wiring layer between each stage is different, which complicates the manufacturing of a device to implement the network. It turns out that by a suitable permutation of the nodes

after each row of swappers, one can simplify the wiring diagram so that all layers becomes identical, as illustrated in Figure 13.3.

The required permutation performed between the wires after each row of swappers, contained within the tall boxes in the figure, turns out to be a so called "perfect shuffle" permutation on N elements, which draws its name from card shuffling. The perfect shuffle (or Faro shuffle) is defined as

$$\begin{cases} j \mapsto 2j & \text{if } j < N/2 \\ j \mapsto 2j + N - 1 & \text{otherwise,} \end{cases}$$

and corresponds to interleaving the top and bottom halves in the input [63].

Figure 13.3 illustrates that if we write the input row j as a binary number composed of $\log_2 N$ bits, a perfect shuffle simply permutes its bits cyclically, shifting them all one notch to the left and moving the leftmost bit all the way to the right. $\log_2 N$ perfect shuffles thus restores j to its original value. Since the rows of swappers in Figure 13.3 can flip the rightmost bit (exchanging two neighboring rows), the net effect of the n^{th} control bit from the left at the top of the figure is thus to control the n^{th} bit from the right of j. In other words, Figure 13.3 corresponds to the same permutation sequence $p(c, i) = c \operatorname{xor} i$ as Figure 13.3 except for the trivial modification that the control variable i has its bits in reverse order.

To further reduce cost, we can omit the last perfect shuffle layer, since the resulting network is equivalent to the butterfly network with its outputs permuted and retains all of the necessary properties.

13.2. Implementation

13.2.1. Layout

Section 13.1 described the basic layout of the butterfly network. We can solve the corner turning problem for an N-element interferometer using an N-link butterfly network. Since this technique is meant to scale to *large* radio telescopes, an ideal implementation of the butterfly network would be built out of large (but not too large) numbers of identical, inexpensive, and easy-to-connect parts.

In the discussion below, we use $n = \log_2 N$ to refer to the number of layers in the network.

13.2.1.1. Network cost

If we built a butterfly network corner turner out of modular components, then the cost could be roughly computed by counting the number of each type of component. To build a very large network, or to build many smaller networks, it would be worth the extra effort to design the components so that they would be inexpensive to manufacture and so that as few as possible would be needed.

The simplest set of components to use would be discrete 2×2 switches and single-link cables. Each controlled swapper layer is $N/2$ identical 2×2 switches arranged in a column and each perfect shuffle layer consists of N cables running between pairs of computers or switches. For a hypothetical $2^{20} = 1048576$-node interferometer, this comes out to 20 layers, for a total of 10,485,760 switches and 20,971,520 cables. This is doable (the interferometer would be expensive enough that this would most likely be only a small part of the cost), but connecting 20 layers of over one million cables without making mistakes would be tedious at best.

13.2.1.2. How to further reduce the cost

There is room for a large improvement in the number of parts needed, though: printed circuit boards containing hundreds of components are inexpensive (in 2009, 12 inch by 14 inch circuit boards can be fabricated for less than $30 each, even in small volumes, assuming that the circuit fits on two layers) and cables are available that can carry many links worth of bandwidth. (Of course, using large cables may only be useful internally – unless multiple source or sink nodes are on the same board, the input and output links must each be on its own cable.)

To optimize the perfect shuffles, we will use a simple property of a cyclic bit shift: an n-bit binary number $b_{n-1}b_{n-2}\ldots b_0$ can be circularly shifted one bit to the left by first circularly shifting the leftmost $n-k$ bits one bit to the left, giving $b_{n-2}\ldots b_{k+1}b_k b_{n-1}b_{k-1}\ldots b_0$ and then shifting the rightmost $k+1$ bits one bit to the left, giving $b_{n-2}\ldots b_{k+1}b_k b_{k-1}\ldots b_0 b_{n-1}$ as desired. (For example, $n=5$ and $k=2$ starts with $b_4b_3b_2b_1b_0$, maps it to $b_3b_2b_4b_1b_0$, and finally maps it to $b_3b_2b_1b_0b_4$.)

If the left sides of 2^{n-k} cables, each carrying 2^k links were lined up such that the first cable (cable 0) carried links $0,\ldots,2^k-1$, the second carried links $2^k,\ldots,2\cdot 2^k-1$, etc, then the leftmost $n-k$ bits of the link number could be circularly shifted one bit to the left by arranging the *cables* into a perfect shuffle. The rightmost $k+1$ bits could be circularly shifted one bit to the left by connecting each pair of adjacent cables (i.e. all of the links sharing the leftmost $n-k-1$ bits after the first circular shift) into a board that had two cable inputs and two cable outputs, with the individual links in the cables arranged into a perfect shuffle. This construction with $n=4$ and $k=2$ is shown in Figure 13.4.

From these two components, using $k=6$ (digital cables with 64 digital links are commercially available), each perfect shuffle layer in a 2^{20}-link butterfly network would require 16,384 cables and 8192 two-cable perfect shufflers. This is a nice saving over using single-link cables, especially in terms of the labor needed to assemble the layers.

By using larger circuit boards, many controlled swappers and many two-cable perfect shufflers could fit on a single circuit board, further cutting the number of parts.

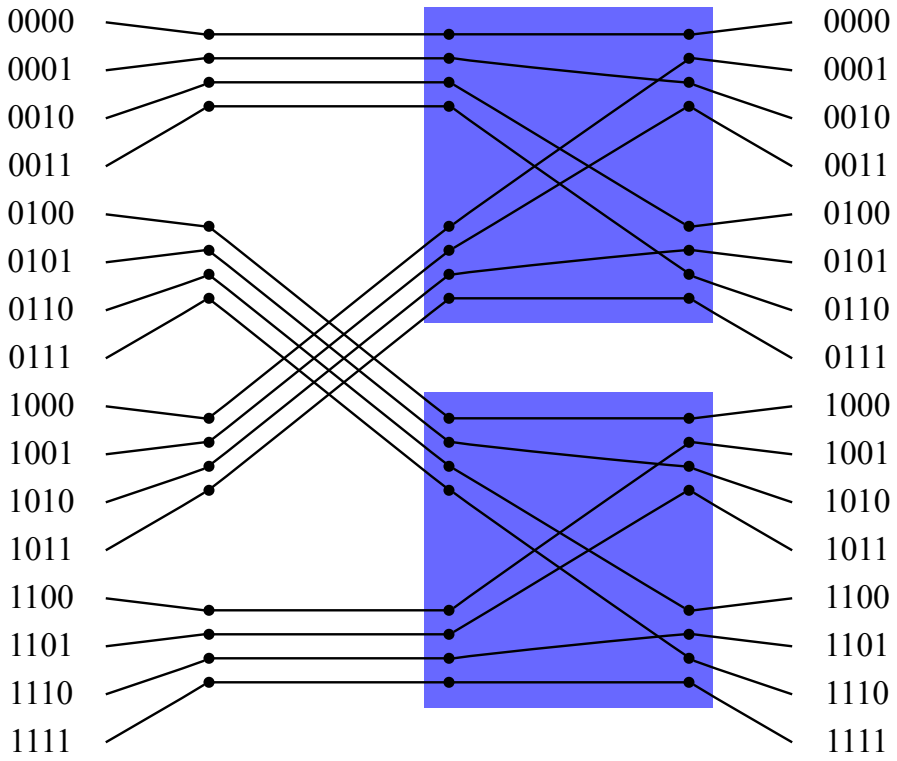


Figure 13.4: A 16-link perfect shuffler built out of four cables, each carrying four links, and two eight-link perfect shufflers, shaded blue.

As an additional improvement, the layers in a butterfly network can be rearranged – rather than using n layers, each of which circularly shifts left by one bit and then xors the least significant bit, the network could instead use n/ℓ layers, each of which xors the least significant ℓ bits with an ℓ-bit control and then circularly shifts by ℓ bits (if n is not a multiple of ℓ, then one layer would involve fewer bits). A device that toggles the least significant ℓ bits of the address does not mix between blocks of 2^ℓ bits, so a simple row of 2^ℓ-link ℓ-bit togglers would make a 2^n-link ℓ bit toggler. This saves a factor of ℓ in the number of togglers needed; ℓ perfect shuffle layers could go between each toggler layer.

Putting this together for a 2^{20}-link butterfly network, using $k = 6$ and $\ell = 8$ gives 20 perfect shuffle layers for a total of 327,680 cables and 163,840 two-cable perfect shufflers and three controlled xor layers containing a total of 12,288 256-link 8-bit togglers (of which one of the three layers would only use half of its control inputs).

Some further improvements would be possible by building layers that circularly shifted by more than one bit at a time – these could still use multiple-link cables

but would need larger, although still probably inexpensive, multiple-bit perfect shuffle boards.

13.2.2. Technology

So far, we have discussed layouts from an abstract point of view, without considering the particular technology used. We will now briefly survey some attractive options for implementing this in practice.

13.2.2.1. Off-the-shelf hardware

The easiest design to prototype would use small (16-port, perhaps) nonblocking Ethernet switches and standard Ethernet cables, with one link per cable. The switches are physically capable of performing any combination of swapping and shuffling on their ports, but they are not generally meant to change their connections on the fly. Most so-called managed switches can, however, store a small number (often 4096) of fixed routes indexed by the destination of the packet that they are switching (this is called the forwarding table). This means that, with some care, the destination field could be used to control the entire route taken by each packet traversing a butterfly network of Ethernet switches. This could be inexpensive (a few tens of dollars per switch port in 2009 for one Gbps per link for a programmable managed switch and significantly more for 10 Gbps) but does not scale well beyond the square root of the size of the forwarding table, giving an upper bound of $N \leq 64$. More flexible switches and routers are available, but tend to be far more expensive and slower.

FPGAs also allow rapid development. Most FPGAs have both standard I/O pins, where a high voltage held for one clock cycle indicates a 1 bit and a low voltage indicates a 0 bit (these pins are limited to relatively low data rates on all but the most expensive FPGAs), and high-speed serial I/O pins, which can send or receive several gigabits per second on differential pairs of pins. Any FPGA can easily act as an arbitrary shuffler or swapper, limited only by the numbers and types of I/O pins it has. We experimented with FPGA implementations and found that a 2×2 controlled swapper, even on a bottom-of-the-line Xilinx FPGA, could switch once per clock cycle, which was 64 million times per second on our device. The downsides of FPGAs are their price and the fact that most FPGAs have relatively few high-speed I/O pins, keeping the cost per link quite high if data rates higher than one bit per clock per link are needed.

13.2.2.2. Digital ASICs

Custom application-specific integrated circuits (ASICs) can be fabricated in 2009 for a few hundred thousand dollars to make a set of masks plus very little for each part produced. Any technology that can transmit and receive high-speed

digital data can also be used to switch it. For example, CMOS devices can switch at moderate speeds (several Gbps) and current-mode devices can operate in excess of 10Gbps per differential pair. The number of links switched on a chip is limited only by the number of pins available on the chip, and a printed circuit board can hold as many of these chips as will fit at very low cost.

In fact, with a signaling technology that can tolerate enough loss, shufflers could be build on circuit boards containing no chips at all, keeping costs even lower.

13.2.2.3. Analog switching

Many protocols for transmitting large data rates over copper cables use advanced modulation techniques. For example, gigabit Ethernet over CAT5 cabling uses multilevel signaling to achieve two gigabits per seconds (1Gbps each way) over four wire pairs at low frequency. Devices to encode, decode, and error correct these kinds of protocols are complex and require significant power to operate, so it would be useful to minimize the number of times that data is modulated and demodulated in a butterfly network. If we used analog switches that could exchange two *modulated* signals with little enough loss, then we could have several layers of controlled swappers between each modulator/demodulator pair.

13.2.2.4. Cable technologies

Technologies to send large data rates over copper cable are well established. Over short distances, a single conductor can carry one link. Over longer distances, differential signals are usually sent over one pair of conductors per link, arranged into some form of transmission line, and different techniques can be used to modulate the signal depending on the frequency response of the transmission line, the available transmission power, and cost considerations. Copper cables have the advantages of being relatively easy to construct, easy to connect, and they can interface easily with switching electronics.

Optical fibers, on the other hand, can carry much higher data rates than copper over a single fiber, and all-optical switching technologies (photonic or mechanical) can rapidly switch these high data rates with little loss. Optical cables are inexpensive, but connecting them is labor-intensive and they are far more expensive to interface with electronics than copper.

Finally, it is possible to transmit very high data rates between boards without any cables at all using free-space optical communication, in which boards have lasers and photodiodes aimed at each other. These laser beams can freely intersect each other, and devices that automatically aim and focus free-space optical links are available. See [64] for an example of a large network switch built out of free-space optical links. This technology could eliminate the need to hand-wire perfect shufflers altogether.

Epilogue: quantum telescopes

In the preface, I said that the two parts of my thesis had nothing in common. That was a bit of a lie.

In Section 8.1, we made the assumption that $kT \gg h\nu$, where T is the brightness temperature of the sky. This assumption allowed us to use the Rayleigh-Jeans approximation to Planck's law. This is convenient but not fundamentally very interesting. It has a much more profound effect, though: $h\nu$ is the energy of a photon and kT is the mean energy per photon mode. So $kT/h\nu$ is the mean number of photons per mode.

If $kT/h\nu \gg 1$, then the light from the sky is bright enough that we can treat it as a classical field, and the two parts of my thesis indeed have nothing in common.

For visible light, $kT/h\nu \lesssim 1$, and the light from the sky is dim enough that quantum corrections matter. In this regime, much of what I said about radio astronomy is incorrect. Most of the incoming modes of light will have zero or one photon. In this regime, light is very much quantum.

A single temporal mode of light covers a small range of wavelengths over a small period of time. Under normal circumstances,[3] each incoming mode will contain a coherent state, and the "antenna response" $A(x, y, \omega, t)$ at some point (x, y) on the ground is the parameter of that coherent state. In the Fock basis, that state is

$$|\psi_{x,y,\omega,t}\rangle = e^{-|A(x,y,\omega,t)|^2/2}\left[|0\rangle + A(x,y,\omega,t)|1\rangle + \frac{1}{2!}A^2(x,y,\omega,t)|2\rangle + \cdots\right]$$

If A is very small, then only the $|0\rangle$ and $|1\rangle$ terms matter, and we can treat each mode as a qubit.

Classical interferometry does not work well in this regime. A normal radio interferometer would try to measure the amplitude and phase of A. If $|A|$ were, for example, 0.1, then the state would be $0.995|0\rangle + 0.0995e^{i\varphi}|1\rangle$, where φ is the phase of that mode. These states are almost the same for different values of φ, so no measurement can reliably distinguish them. An interferometer needs to measure phase, so it would be incredibly noisy unless it used quantum measurements.

The simple solution is to physically interfere the light from two different positions on the ground. This is tricky when those positions are far apart, and quantum computing-inspired tricks have been proposed to improve the process [65].

[3] Highly advanced space aliens sending evenly-spaced photons or squeezed light would be an exception.

Some day, though, visible-light interferometers could work very differently. The telescope could be a array of N quantum antennas. For every mode (ω, t), each antenna (x, y) would collect the qubit $|\psi_{x,y,\omega,t}\rangle$. Instead of measuring these qubits, the antennas would send them on to a quantum computer. That computer would see all of the qubits, and it would process them separately for each (ω, t). For each (ω, t), the quantum computer has the state $|\Psi\rangle = \otimes_{x,y}|\psi_{x,y}\rangle$. The computer would do one of a few things.

If $N|A|^2 \ll 1$, then the state $|\Psi\rangle$ most likely contains zero photons, and two or more photons are very unlikely. The computer would count the number of photons and throw away the state if there is any number other than one. Otherwise the computer would unitarily convert $|\Psi\rangle$ into a quantum register storing the position of the single received photon. If the antennas were on an evenly-spaced grid, it would take the quantum Fourier transform of that register and measure the result. The result would tell the computer which direction the photon came from, and the computer could add a dot to a map of the sky. After accumulating enough photons, the computer would have a picture. This would be a QFTT or Quantum Fourier Transform Telescope. This is almost the same calculation that the mirrors in today's telescopes perform. If the sky were brighter or N were larger, the computer could use the two-photon case as well, and fancier measurements would be possible at the cost of increased quantum computing effort.

We are nowhere near having the technology to build these quantum telescopes. Some day, though, they could revolutionize optical astronomy and give us unimaginably detailed pictures of far-away objects in the sky.

Bibliography

[1] S. Aaronson, E. Farhi, D. Gosset, A. Hassidim, J. Kelner, and A. Lutomirski, "Quantum money." Unpublished review article.

[2] E. Farhi, D. Gosset, A. Hassidim, A. Lutomirski, and P. Shor, "Quantum money from knots," arXiv:1004.5127 [quant-ph].

[3] A. Lutomirski, S. Aaronson, E. Farhi, D. Gosset, A. Hassidim, J. Kelner, and P. Shor, "Breaking and making quantum money: toward a new quantum cryptographic protocol," in *Innovations in Computer Science*. 2010. arXiv:0912.3825v1 [quant-ph]. http://conference.itcs.tsinghua.edu.cn/ICS2010/content/papers/2.html.

[4] A. Lutomirski, "Component mixers and a hardness result for counterfeiting quantum money," arXiv:1107.0321v1 [quant-ph].

[5] E. Farhi, D. Gosset, A. Hassidim, A. Lutomirski, D. Nagaj, and P. Shor, "Quantum state restoration and single-copy tomography for ground states of hamiltonians," *Phys. Rev. Lett.* **105** no. 19, (Nov, 2010) 190503.

[6] A. Lutomirski, "An online attack against wiesner's quantum money," arXiv:1010.0256v1 [quant-ph].

[7] A. Lutomirski, M. Tegmark, N. J. Sanchez, L. C. Stein, W. L. Urry, and M. Zaldarriaga, "Solving the corner-turning problem for large interferometers," *Monthly Notices of the Royal Astronomical Society* **410** no. 3, (2011) 2075–2080.

[8] M. T. Bixler, *Winds of Freedom: The Story of the Navajo Code Talkers of World War II.* Two Bytes Pub. Co., Darien, CT, 1992.

[9] W. Diffie and M. Hellman, "New directions in cryptography," *Information Theory, IEEE Transactions on* **22** no. 6, (1976) 644–654.

[10] A. C. Yao, "Protocols for secure computations," *Foundations of Computer Science, Annual IEEE Symposium on* (1982) 160–164.

[11] C. Bennett, G. Brassard, *et al.*, "Quantum cryptography: Public key distribution and coin tossing," in *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing*, vol. 175, Bangalore, India. 1984. http://www.research.ibm.com/people/b/bennetc/bennettc198469790513.pdf.

[12] C. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal* **28** no. 4, (October, 1949) 656–715. http://www.alcatel-lucent.com/bstj/vol28-1949/articles/bstj28-4-656.pdf.

[13] M. N. Wegman and L. Carter, "New hash functions and their use in authentication and set equality," *Journal of Computer and System Sciences* **22** (1981) 265–279.

[14] R. Ryan, Z. Anderson, and A. Chiesa, "The anatomy of a subway hack," in *DEFCON 16*. August, 2008. http://tech.mit.edu/V128/N30/subway/Defcon_Presentation.pdf.

[15] S. Wiesner, "Conjugate coding," *SIGACT News* **15** no. 1, (1983) 78–88.

[16] C. Bennett, G. Brassard, S. Breidbart, and S. Wiesner, "Quantum cryptography, or unforgeable subway tokens," in *Advances in Cryptology–Proceedings of Crypto*, vol. 82, pp. 267–275. 1983.

[17] H.-K. Lo, "Insecurity of quantum secure computations," *Phys. Rev. A* **56** no. 2, (Aug, 1997) 1154–1162.

[18] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.* **26** no. 5, (1997) 1484–1509.

[19] Y. Tokunaga, T. Okamoto, and N. Imoto, "Anonymous quantum cash," in *EQIS*. August, 2003. http://www.qci.jst.go.jp/eqis03/program/papers/009-Tokunaga.ps.gz.

[20] S. Aaronson, "Quantum copy-protection and quantum money," in *Computational Complexity, Annual IEEE Conference on*, pp. 229–242. 2009.

[21] M. Mosca and D. Stebila, "Quantum coins," in *Error-Correcting Codes, Finite Geometries, and Cryptography*, vol. 523 of *Contemporary Mathematics*, pp. 35–47. American Mathematical Society, 2010. arXiv:0911.1295v1 [quant-ph].

[22] R. Merkle, "A digital signature based on a conventional encryption function," in *Advances in Cryptology — CRYPTO '87*, C. Pomerance, ed., vol. 293 of *Lecture Notes in Computer Science*, pp. 369–378. Springer Berlin / Heidelberg, 2006.

[23] C. H. Bennett, "Quantum cryptography: Uncertainty in the service of privacy," *Science* **257** no. 5071, (1992) 752–753.

[24] C. Marriott and J. Watrous, "Quantum arthur-merlin games," *Computational Complexity* **14** no. 2, (2005) 122–152.

[25] D. Nagaj, P. Wocjan, and Y. Zhang, "Fast amplification of QMA," *Quantum Information & Computation* **9** no. 11&12, (2009) 1053–1068, `arXiv:0904.1549v1 [quant-ph]`.

[26] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association* **58** no. 301, (1963) 19–30.

[27] C. Jordan, "Essai sur la géométrie à n dimensions," *Bulletin de la S. M. F.* **3** (1875) 103–174.

[28] M. A. Nielsen and I. L. Chuang, *Quantum Information and Computation*. Cambridge University Press, Cambridge, UK, 2000.

[29] D. Poulin and P. Wocjan, "Preparing ground states of quantum many-body systems on a quantum computer," *Physical Review Letters* **102** no. 13, (2009) 130503.

[30] D. Zuckerman, "Linear degree extractors and the inapproximability of max clique and chromatic number," *Theory of Computing* **3** no. 1, (2007) 103–128.

[31] N. Alon, M. Krivelevich, and B. Sudakov, "Finding a large hidden clique in a random graph," *Random Structures & Algorithms* **13** (October, 1998) 457–466.

[32] U. Feige and R. Krauthgamer, "Finding and certifying a large hidden clique in a semirandom graph," *Random Structures & Algorithms* **16** no. 2, (2000) 195–208.

[33] L. Grover and T. Rudolph, "Creating superpositions that correspond to efficiently integrable probability distributions," `arXiv:quant-ph/0208112`.

[34] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, "Strengths and weaknesses of quantum computing," *SIAM J. Comput.* **26** (October, 1997) 1510–1523.

[35] J. Watrous, "Succinct quantum proofs for properties of finite groups," *Foundations of Computer Science, Annual IEEE Symposium on* (2000) 537–456.

[36] S. Aaronson and G. Kuperberg, "Quantum versus classical proofs and advice," *Theory of Computing* **3** no. 1, (2007) 129–157.

[37] L. Babai, "Local expansion of vertex-transitive graphs and random generation in finite groups," in *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, STOC '91, pp. 164–174. ACM, New York, NY, USA, 1991.

[38] A. Sahai and S. Vadhan, "A complete problem for statistical zero knowledge," *J. ACM* **50** (March, 2003) 196–249.

[39] J. W. Alexander, "Topological invariants of knots and links," *Transactions of the American Mathematical Society* **30** no. 2, (1928) 275–306. http://www.jstor.org/stable/1989123.

[40] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller, *et al.*, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics* **21** no. 6, (1953) 1087.

[41] D. Aharonov and A. Ta-Shma, "Adiabatic quantum state generation," *SIAM Journal on Computing* **37** no. 1, (2007) 47–82.

[42] J. Hass and T. Nowik, "Unknot diagrams requiring a quadratic number of Reidemeister moves to untangle," *Discrete & Computational Geometry* **44** (2010) 91–95.

[43] L. von Ahn, M. Blum, N. Hopper, and J. Langford, "Captcha: Using hard ai problems for security," in *Advances in Cryptology — EUROCRYPT 2003*, E. Biham, ed., vol. 2656 of *Lecture Notes in Computer Science*, pp. 646–646. Springer Berlin / Heidelberg, 2003.

[44] M. Bellare and P. Rogaway, "Optimal asymmetric encryption," in *Advances in Cryptology — EUROCRYPT'94*, A. De Santis, ed., vol. 950 of *Lecture Notes in Computer Science*, pp. 92–111. Springer Berlin / Heidelberg, 1995.

[45] A. Ambainis, L. Magnin, M. Roetteler, and J. Roland, "Symmetry-assisted adversaries for quantum state generation," in *Computational Complexity (CCC), 2011 IEEE 26th Annual Conference on*, pp. 167–177. June, 2011.

[46] D. Larson, J. Dunkley, G. Hinshaw, E. Komatsu, M. R. Nolta, C. L. Bennett, B. Gold, M. Halpern, R. S. Hill, N. Jarosik, A. Kogut, M. Limon, S. S. Meyer, N. Odegard, L. Page, K. M. Smith, D. N. Spergel, G. S. Tucker, J. L. Weiland, E. Wollack, and E. L. Wright, "Seven-year wilkinson microwave anisotropy probe (wmap) observations: Power spectra and wmap-derived parameters," *The Astrophysical Journal Supplement Series* **192** no. 2, (2011) 16.

[47] M. Tegmark and M. Zaldarriaga, "Fast Fourier transform telescope," *Phys. Rev. D* **79** no. 8, (April, 2009) 083530.

[48] A. Parsons, M. McQuinn, D. Jacobs, J. Aguirre, and J. Pober, "A sensitivity and array-configuration study for measuring the power spectrum of 21cm emission from reionization," `arXiv:1103.2135v1 [astro-ph]`.

[49] A. Thompson, J. Moran, and G. Swenson, *Interferometry and Synthesis in Radio Astronomy*. Wiley-VCH, 2nd ed., 2001.

[50] J. Pritchard and A. Loeb, "Cosmology: Hydrogen was not ionized abruptly," *Nature* **468** (2010) 772–773.

[51] E. Otobe, J. Nakajima, K. Nishibori, T. Saito, H. Kobayashi, N. Tanaka, N. Watanabe, Y. Aramaki, T. Hoshikawa, K. Asuma, and D. T., "Two-dimensional direct images with a spatial FFT interferometer," *Publications of the Astronomical Society of Japan* **46** (1994) 503–510. `http://adsabs.harvard.edu/full/1994PASJ...46..503O`.

[52] M. Tegmark and M. Zaldarriaga, "Omniscopes: Large area telescope arrays with only $N \log N$ computational cost," *Phys. Rev. D* **82** no. 10, (November, 2010) 103501.

[53] N. J. Sanchez, "On the Instrumentation of the Omniscope," Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, June, 2011.

[54] A. Liu, M. Tegmark, S. Morrison, A. Lutomirski, and M. Zaldarriaga, "Precision calibration of radio interferometers using redundant baselines," *Monthly Notices of the Royal Astronomical Society* **408** no. 2, (2010) 1029–1050.

[55] A. N. Perko, "The Omniscope: Mapping the universe in 3D with neutral hydrogen." Bachelor's thesis, Massachusetts Institute of Technology, Cambridge, MA, June, 2011.

[56] "Center for astronomy signal processing and electronics research." `http://casper.berkeley.edu`.

[57] A. Parsons, D. Backer, A. Siemion, H. Chen, D. Werthimer, P. Droz, T. Filiba, J. Manley, P. McMahon, A. Parsa, D. MacMahon, and M. Wright, "A scalable correlator architecture based on modular FPGA hardware, reuseable gateware, and data packetization," *Publications of the Astronomical Society of the Pacific* **120** no. 873, (November, 2008) 1207–1221.

[58] A. R. Parsons, D. C. Backer, G. S. Foster, M. C. H. Wright, R. F. Bradley, N. E. Gugliucci, C. R. Parashare, E. E. Benoit, J. E. Aguirre, D. C. Jacobs, C. L. Carilli, D. Herne, M. J. Lynch, J. R. Manley, and D. J. Werthimer, "The precision array for probing the epoch of re-ionization: Eight station results," *The Astronomical Journal* **139** no. 4, (2010) 1468.

[59] L. R. D'Addario, "Generalization of the memoryless corner turner to the non-square case." *Allen Telescope Array* memo no. 22, April, 2001. http://ral.berkeley.edu/ata/memos/memo22.pdf.

[60] W. L. Urry, "A corner turner architecture." *Allen Telescope Array* memo no. 14, November, 2000. http://ral.berkeley.edu/ata/memos/memo14.pdf.

[61] W. L. Urry, M. C. H. Wright, M. Dexter, and D. MacMahon, "The ATA correlator." *Allen Telescope Array* memo no. 73, February, 2007. http://ral.berkeley.edu/ata/memos/memo73.pdf.

[62] A. Youssef and B. Arden, "Topology of efficiently controllable banyan multistage networks," *Microprocessors and Microsystems* **16** no. 1, (1992) 3–13.

[63] H. Stone, "Parallel processing with the perfect shuffle," *Computers, IEEE Transactions on* **C-20** no. 2, (February, 1971) 153–161.

[64] K. Hirabayashi, T. Yamamoto, and M. Yamaguchi, "Free-space optical interconnections with liquid-crystal microprism arrays," *Appl. Opt.* **34** no. 14, (May, 1995) 2571–2580.

[65] D. Gottesman, T. Jennewein, S. Croke, and L. Boyle, "Longer baseline telescope arrays using quantum repeaters," in *International Conference on Quantum Information*, p. PDPB1. Optical Society of America, 2011. http://www.opticsinfobase.org/abstract.cfm?URI=ICQI-2011-PDPB1.