Lecture 6 The Shor Code & Basic Benchmarking

§ The Shor code

(a) QPU's have more exotic kinds of errors than CPU's. While CPU's can only have (potentially correllated) bit flip errors, each qubit has errors that can be caused by any of the continuous family of rotations that make up transformations on the Bloch sphere. Further, quantum errors can not only be correllated like classical errors, but also they can be entangled errors.

(b) It is hard to check for errors in quantum memory. Because any measurement changes the state of quantum memory, we need to be clever about how we look at the memory to determine if an error has occurred.

(c) We cannot blindly duplicate states due to the no-cloning theorem. Thus we need to be clever about how we build redundancy into our codes.

Given (a) + (b) + (c) it is amazing that quantum error correction is possible at all!

Thm [No-cloning]

There exists no unitary U s.t. for arbitrary quantum states $|\psi\rangle$ and $|s\rangle$

$$U(|\psi\rangle \otimes |s\rangle) = |\psi\rangle \otimes |\psi\rangle$$

Proof (by contradiction) Recall that $(A\otimes B)(C\otimes D) = AC \otimes BD$ for the tensor product.

Suppose $|\psi\rangle, |s\rangle$ and $|\varphi\rangle$ are arbitrary states and so

$$U(|\psi\rangle\otimes|s\rangle) = |\psi\rangle\otimes|\psi\rangle \quad \text{AND} \quad U(|\varphi\rangle\otimes|s\rangle) = |\varphi\rangle\otimes|\varphi\rangle$$

Then $\Rightarrow$ $\langle\psi|\varphi\rangle = \langle\psi|\varphi\rangle\langle s|s\rangle = (\langle\psi|\otimes\langle s|)U^{\dagger}U(|\varphi\rangle\otimes|s\rangle)$

$$= (\langle\psi|\otimes\langle\psi|)(|\varphi\rangle\otimes|\varphi\rangle) = (\langle\psi|\varphi\rangle)^2$$

But $x = x^2$ has only two cases:

$$\langle\psi|\varphi\rangle = 0 \Rightarrow |\psi\rangle \text{ is orthogonal to } |\varphi\rangle$$

or

$$\langle\psi|\varphi\rangle = 1 \Rightarrow |\psi\rangle = |\varphi\rangle$$

Thus one can only clone states that are orthogonal to one another.

This is why CNOT can map $|x,0\rangle \to |x,x\rangle$ as $|0\rangle$ and $|1\rangle$ are orthogonal.

Note This rules out "perfect" copying. Approximate copying is possible.

Recap [Parity measurements to correct errors]

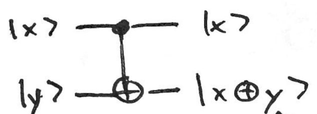
eigenvector	eigenvalue
$\|00\rangle$	1
$\|11\rangle$	1
$\|01\rangle$	-1
$\|10\rangle$	-1

Recall $Z_1 Z_0$ checks the parity of qubits 0 & 1, because $Z_1 \otimes Z_0$ has

Thus measuring $Z_1 Z_0$ returns ± 1 depending on the parity.

Q: But in Quil how do we do this! A: With CNOTs.

Note that



← the parity of x and y.

Thus to measure the parity of a set of qubits we use

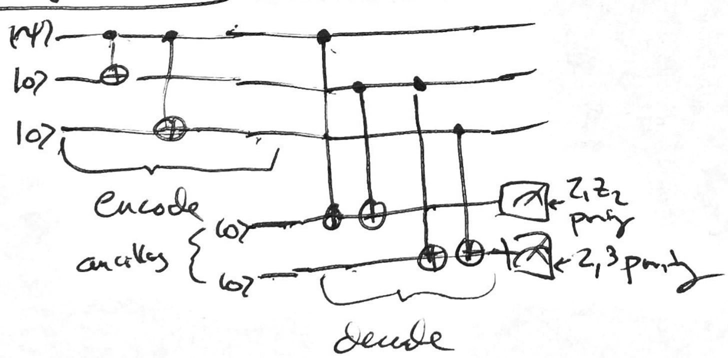
Ancilla a $\overset{\text{in state}}{=}$ $|0\rangle$ to start.

for x in parity-set:
CNOT x a
MEASURE a ro[0] ← this now stores the joint parity

Note that while parity is a useful global property (as we will see) it tells us literally nothing about the actual values of our qubits. This is important and is why it has no effect on the computational states.

Recap

Bit flip code



In Quil Let x be data qubit index
x_1, x_2 be encoder indices
a_1, a_2 ancilla indices

CNOT x x_1
CNOT x x_2 } encode
← errors
CNOT x a_1
CNOT x_1 a_1 } decode parity $x \oplus x_1$
CNOT x_1 a_2
CNOT x_2 a_2
MEASURE a_1 ro[0] ← $x \oplus x_1$
MEASURE a_2 ro[1] ← $x_1 \oplus x_2$

Classical memory

ro[0]	ro[1]	correction
0	0	none
1	0	X x
0	1	X x_2
1	1	X x_1

} recovery

Phase Flip code

So we can detect bit flips, but what about other errors.

For example how do we detect "phase flips" $a|0\rangle + b|1\rangle \longmapsto a|0\rangle - b|1\rangle$

error type	bit errors	phase errors
modelled as	random X gates	random Z gates

Since we already know how to correct bit flips, we need to convert phase flips into bit flips. This is done w/ the Hadamard.

Thm $HZH = X$

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

Thus prepending and appending a H can convert a phase flip to a bit flip.

another example: $H(a|0\rangle + b|0\rangle) = \frac{a+b}{\sqrt{2}}|0\rangle + \frac{a-b}{\sqrt{2}}|1\rangle \xmapsto{b \mapsto -b} \frac{a-b}{\sqrt{2}}|0\rangle + \frac{a+b}{\sqrt{2}}|1\rangle$

Looks like a bit flip.

Phase flip code in Quil

```
CNOT x x1
CNOT x x2
H x
H x1
H x2
- - - - - - -   ← errors
H x
H x1
H x2
CNOT x a1
CNOT x1 x1
CNOT x1 a2
CNOT x2 a2
MEASURE a1 ro[0]
MEASURE a2 ro[1]
```

encode

Identity $H^2 = I$

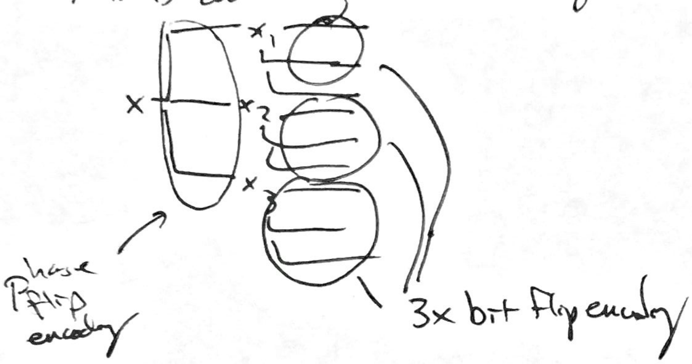
decode

ro[0] ro[1] tells us where to apply $\underline{\underline{Z}}$ gates to correct the phase errors

The 9-qubit Shor code

But we need to correct Both phase and bit flip errors.

This is done by "concatenating" the bit and phase codes.



→ The idea is to use the phase code to encode into 3 qubits and then for each of those qubits to encode into the bit flip code

In Quil

```
DEFCIRCUIT BIT_ENCODE q0 q1 q2:
    CNOT q0 q1
    CNOT q0 q2
```

~~DEFCIRCUIT BIT_DECODE q0 q1 q2 ...~~

```
DEFCIRCUIT PARITY list[q] a r:
    for qi in list[q]:
        CNOT qi a
    MEASURE a ro[r]
```

```
BIT_ENCODE x x1 x2
H x, x1, x2          # shorthand for 3 hadamards
BIT_ENCODE x xA xB
BIT_ENCODE x1 x1A x1B
BIT_ENCODE x2 x2A x2B
PARITY [x, xA] a0 ; PARITY [xA, xB] a0'
PARITY [x1, x1A] a1 ; PARITY [x1A, x1B] a1'      } to find bit errors
PARITY [x2, x2A] a2 ; PARITY [x2A, x2B] a2'
H x, xA, xB, x1, x1A, x1B, x2, x2A, x2B
PARITY [x, xA, xB, x1, x1A, x1B] b0              } to find phase errors
PARITY [x1, x1A, x1B, x2, x2A, x2B] b1
```

Deep fact!

In fact the Shor code can correct arbitrary single qubit errors.

This is fascinating because it is an example where a continuum of errors can be corrected by a discrete subset of those errors.

Recall that any ~~Kraus operator~~ arbitrary error is given by

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger$$

And any E_i can be written as:

$$E_i = \alpha I + \beta X + \gamma Z + \delta XZ$$

Thus $E_i |\psi\rangle$ is a superposition over $\begin{cases} \text{no error} \\ \text{bit error} \\ \text{phase error} \\ \text{both} \end{cases}$

Measuring the error syndrome (making the parity measurements) collapses this superposition! This allows discrete correction.

The basic idea is that the same reason:

$a|0\rangle + b|1\rangle$ looks continuous but when measured is $\{0, 1\}$
allows continuous errors to be measured to allow discrete correction.

The Shor code is an example of a $[9, 1, 3]$ code:

9 physical qubits
1 logical qubit
3 $\underbrace{\text{code distance}}_{\text{Hamming distance between code words}}$ $(3-1)/2 = 1$ arbitrary error corrected

More efficient $[7, 1, 3]$ and $[5, 1, 3]$ codes exist.

We also have only considered: (so far)

(1) Single qubit errors
(2) independent errors
(3) acting as a memory rather than protecting real computation.

Benchmarking

Given that QPU's have noise, how do you characterize what QPU you have? How do you tell if its good enough to be better than CPUs?

Levels of benchmarking

2: algorithm (app end-to-end benchmarks)

1: device (quantum FLOPS)

0: physical (basic operations) ← This lecture

How do you tell if your operators did what you wanted?

Metrics → Inner product $\langle\psi|\phi\rangle$ but this only works for pure states

→ State Fidelity $F(\rho,\sigma) = tr(\rho^{1/2}\sigma\rho^{1/2})$

Facts

→ bounded $0 \le F(\rho,\sigma) \le 1$

→ $F(\rho,\sigma) = 1 \Leftrightarrow \rho = \sigma$

→ symmetric $F(\rho,\sigma) = F(\sigma,\rho)$ ← This is non-trivial to show Thm 9.4 Mike and Ike

→ unitary invariant $F(U\rho U^{+}, U\sigma U^{+}) = F(\rho,\sigma)$

→ monotonic $F(\mathcal{E}(\rho),\mathcal{E}(\sigma)) \ge F(\rho,\sigma)$ ← Thm 9.5 Mike & Ike

→ Ulmann's thm $F(\rho,\sigma) = \max_{\psi,\varphi} |\langle\psi|\varphi\rangle|^2$ ← purifications of ρ and σ respectively

If we are comparing to "pure" states then the fidelity simplifies to:

$$F(|\psi\rangle,\rho) = \sqrt{\langle\psi|\rho|\psi\rangle}$$

An Aside on Density Matrix measurement

To Benchmark mixed states we need to understand how to perform measurements on them.

Let A be an observable {e.g. a Pauli X, Y, or Z} and let there be an ensemble of states $|\Psi_a\rangle$ for our system that each occur w/ probability p_a.

The expectation of A with respect to the ensemble is then:

$$\sum p_a \langle\Psi_a|A|\Psi_a\rangle = \sum p_a \, tr\left[\langle\Psi_a|A|\Psi_a\rangle\right]$$

$$= \sum p_a \, tr\left[A|\Psi_a\rangle\langle\Psi_a|\right] \quad \text{as traces are cyclic } tr(ABC) = tr(CAB)$$

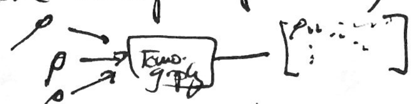
$$= tr\left[A\sum p_a |\Psi_a\rangle\langle\Psi_a|\right]$$

$$= tr(A\rho)$$

Thus the expectation of a density matrix repr of a state for some observable is given by $tr(A\rho)$.

State Tomography

So to measure fidelity we need to prepare states and measure their density matrices ρ. We must have multiple copies of ρ in order to charaterize it.



Suppose we have multiple copies of ρ (or can make them on demand).

We can expand $$\rho = \frac{tr(\rho) I + tr(X\rho)X + tr(Y\rho)Y + tr(Z\rho)Z}{2}$$

I, X, Y, Z form a basis for 2×2 complex matrices

$tr(X\rho)$ is $\langle X \rangle_\rho$ i.e. the expectation value of ρ when measured by the X observable.

Thus to get ρ we must calculate $\langle X\rangle_\rho$, $\langle Y \rangle_\rho$, $\langle Z\rangle_\rho$ (the 4th from normalization)

$\uparrow \quad \uparrow \quad \uparrow$

3 sets of code

- MEASURE q ro[0] ← measures in the Z observable
- prepending a $RY(-\pi/2)$ q will measure in X • $RX(\pi/2)$ q measures in Y
- ~~[illegible]~~

Run each say 1000 times and calculate the expected value

bit	value
$0 \mapsto$	-1
$1 \mapsto$	1

This scales badly! For n-qubits you need $(2^{n^2}-1)$ types of observables

It works well but takes time.

There is also Process Tomography that is similar but characterizes a channel $\mathcal{E}$.

Readout Fidelity

I 0 vs X 0

MEASURE 0 ro[0] MEASURE 0 ro[0]

What % of the time do we get the right answer?

Tricky because gates are also imperfect. More sophisticated techniques decouple these.

T1 ~~#~~ coherence

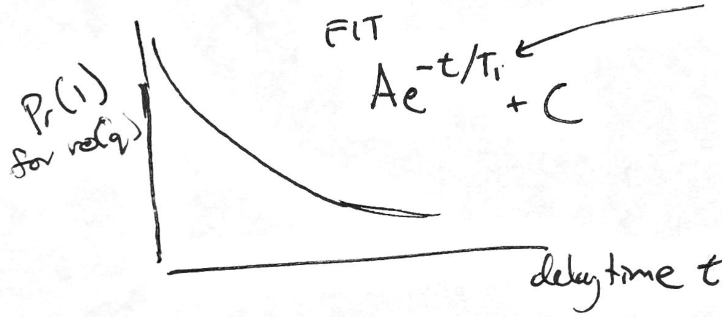
Sometimes we want more synthetic measures of performance.
T1 measures the impact of amplitude damping.

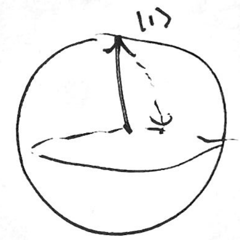
Idea:

X q
delay for t time ← the longer you wait, the ~~longer the~~ higher the chance of decoherence.
MEASURE q ro[q]

FIT ← the characteristic decay rate is T1

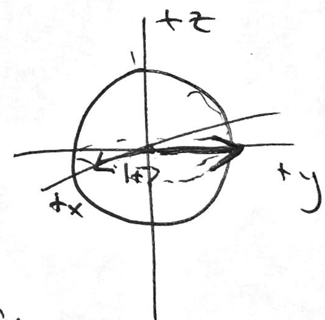
$$Ae^{-t/T_1} + C$$



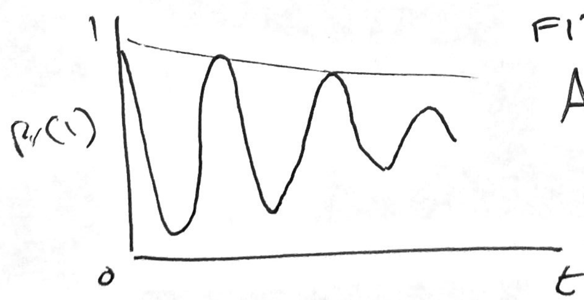


T2 coherence

This is the same but phase shifted.



RX(π/2) q
wait for t ← we add a "detuning" here to make it easier to fit RZ(2πtω_d)
RX(π/2) q
MEASURE q ro[q]



FIT ← T2 time

$$Ae^{-2t/T_2}\sin(\omega_d(t-\cancel{\phi}))+c$$

How many steps?
How many samples?
How to fit?