CPEN 400Q

Variational **G**raphical **Q**uantum **E**rror **C**orrection

Group 8: Lina Bekken Babusiaux Kaushik Kolla Richard Nguyen

Table of Contents

I. Introduction

What’s the problem?

II. VGQEC

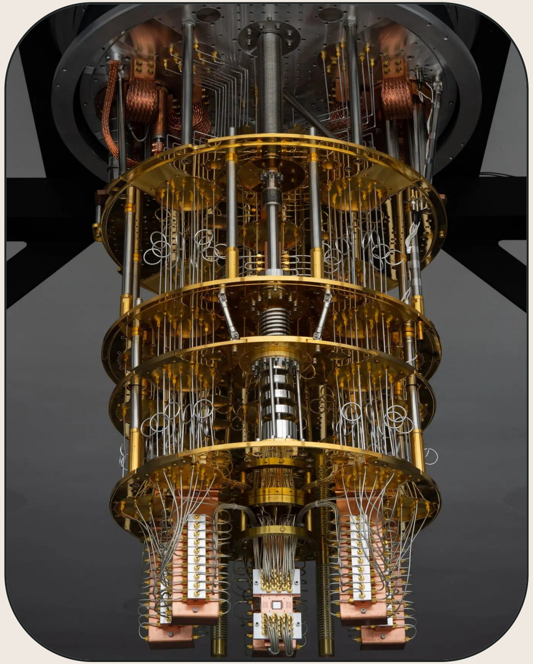
Any solutions?

III. Implementation

How is it implemented?

IV. Results

Did it work?



I.

Introduction

Current Strides in Error Correction

- ➔ Google’s Willow (Dec 2024)
  - ◆ 105 qubits
  - ◆ Demonstrated error rates decrease as code size increases.

Basic types of Error Correction

Repetition

❖ Simplest error correction strategy

❖ Can only correct only one type of error (X or Z, not both)

❖ Detects errors using majority voting

➢ Requires many physical qubits per logical qubit

➢ Cannot correct both bit and phase-flips simultaneously

Stabilizers - [[5,1,3]] code

❖ Smallest code that detects and corrects all single-qubit errors (X, Y, Z)

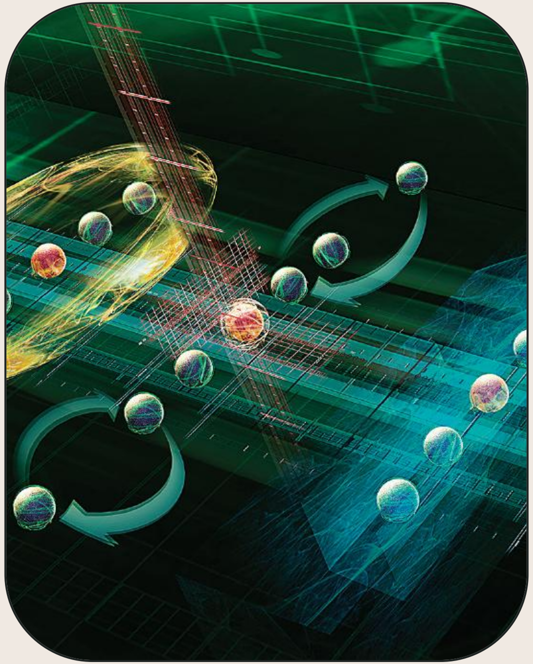
❖ 1 Logical qubit ⟶ 5 physical qubits

❖ High efficiency

➢ Complex

➢ Requires non-local stabilizers

➢ Syndrome decoding is non-trivial compared to repetition



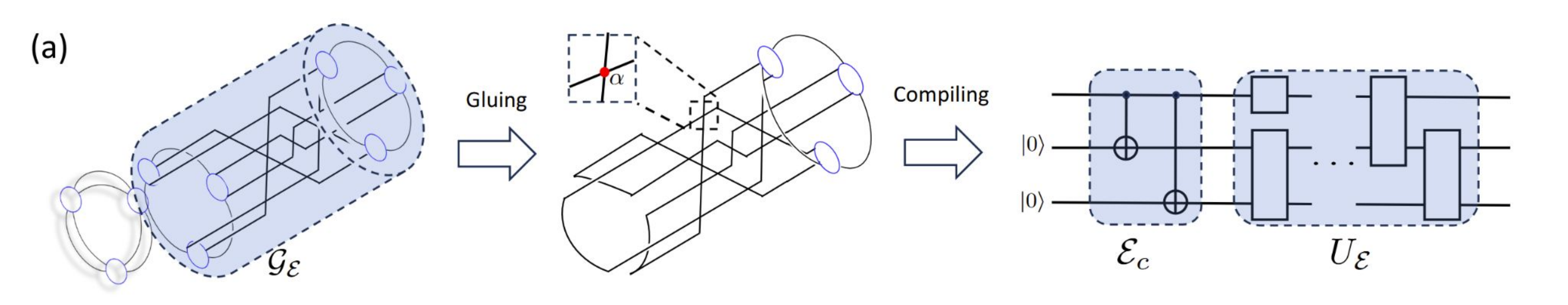
II.

VGQEC

Hybrid Quantum-Classical scheme for VGQEC (12 Mar 2025) [1]

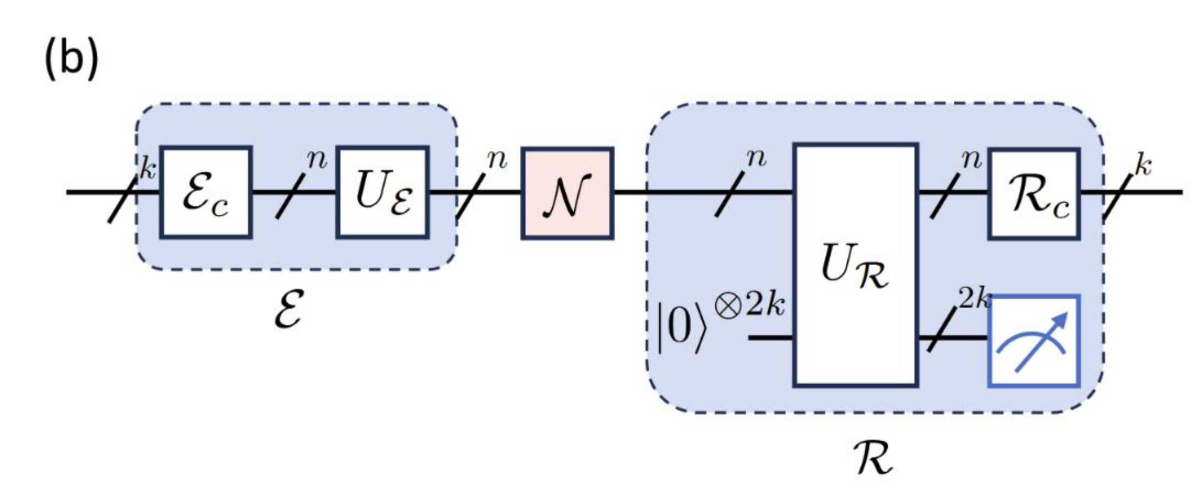
[1] https://arxiv.org/abs/2410.02608

VGQEC



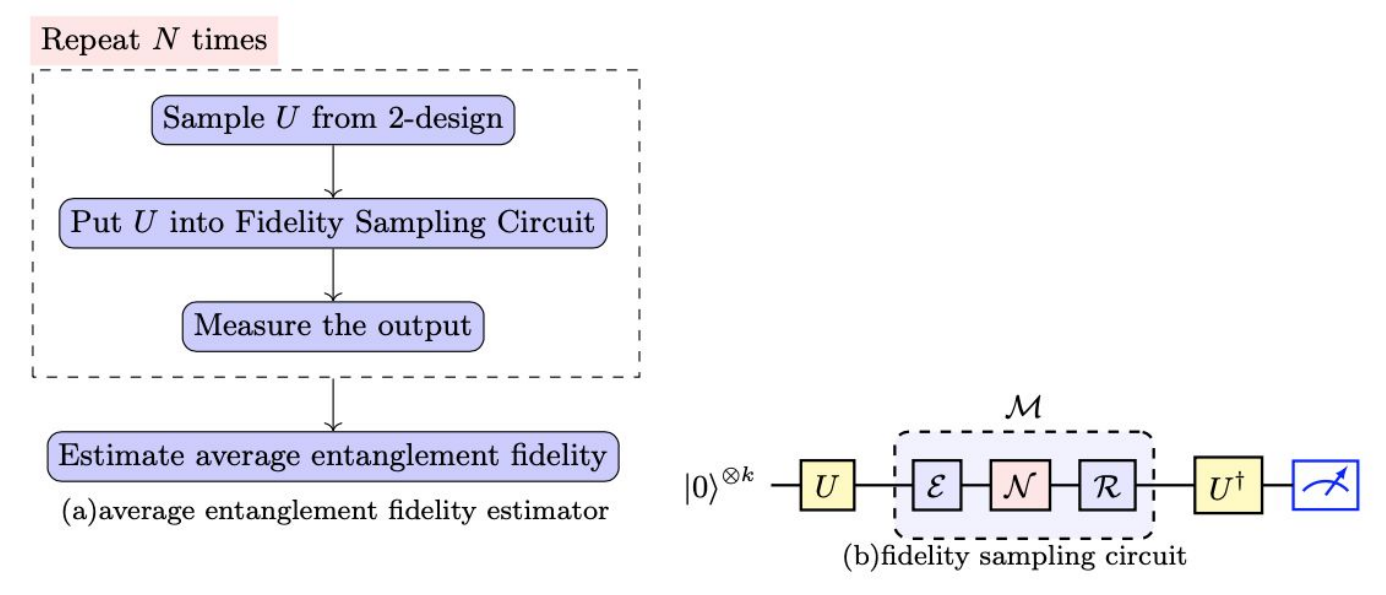
a) Build new code by adding a flexible graph (GE) to existing code’s QEC Quon graph. Creates new encoding circuit with tunable parameters.

VGQEC



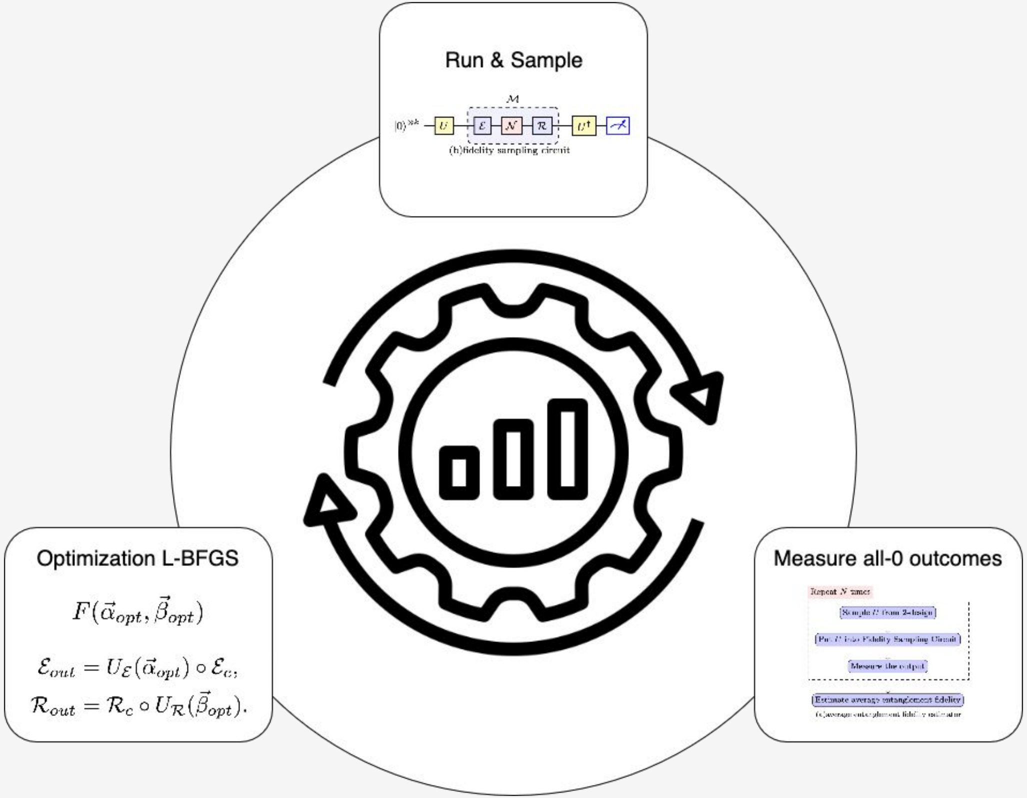
b) Run noisy circuit using encoding and recovery steps. Recovery circuit uses extra helper (auxiliary) qubits and measurements to correct noise.

VGQEC



a) Repeatedly run circuit with random quantum state and count output of all zeros.
b) Create random quantum state, send through noise, reverse randomness, and measure.

VGQEC



Run & Sample

(h)fidelity sampling circuit

Measure all-0 outcomes

Repeat N times

Sample U from 2-design

Put U into Fidelity Sampling Circuit

Measure the output

Estimate average entanglement fidelity

(i)average entanglement fidelity estimator

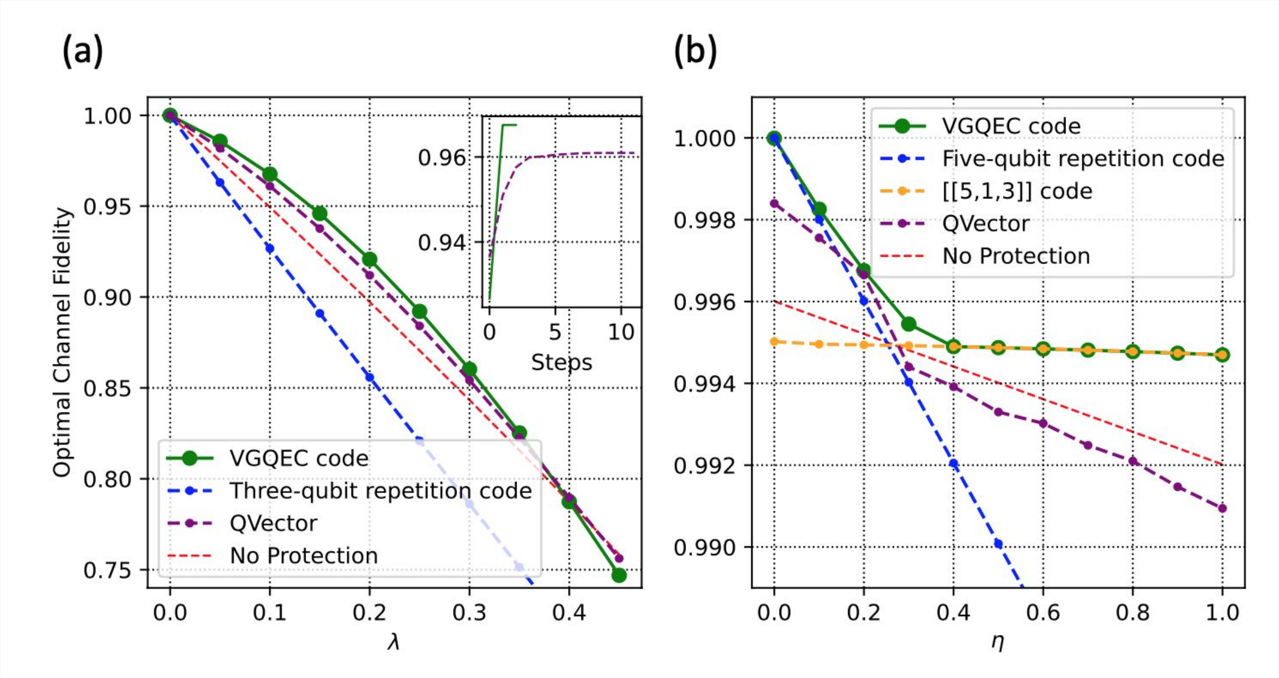
Optimization L-BFGS

$$F(\vec{\alpha}_{opt}, \vec{\beta}_{opt})$$

$$\mathcal{E}_{out} = U_{\mathcal{E}}(\vec{\alpha}_{opt}) \circ \mathcal{E}_c,$$

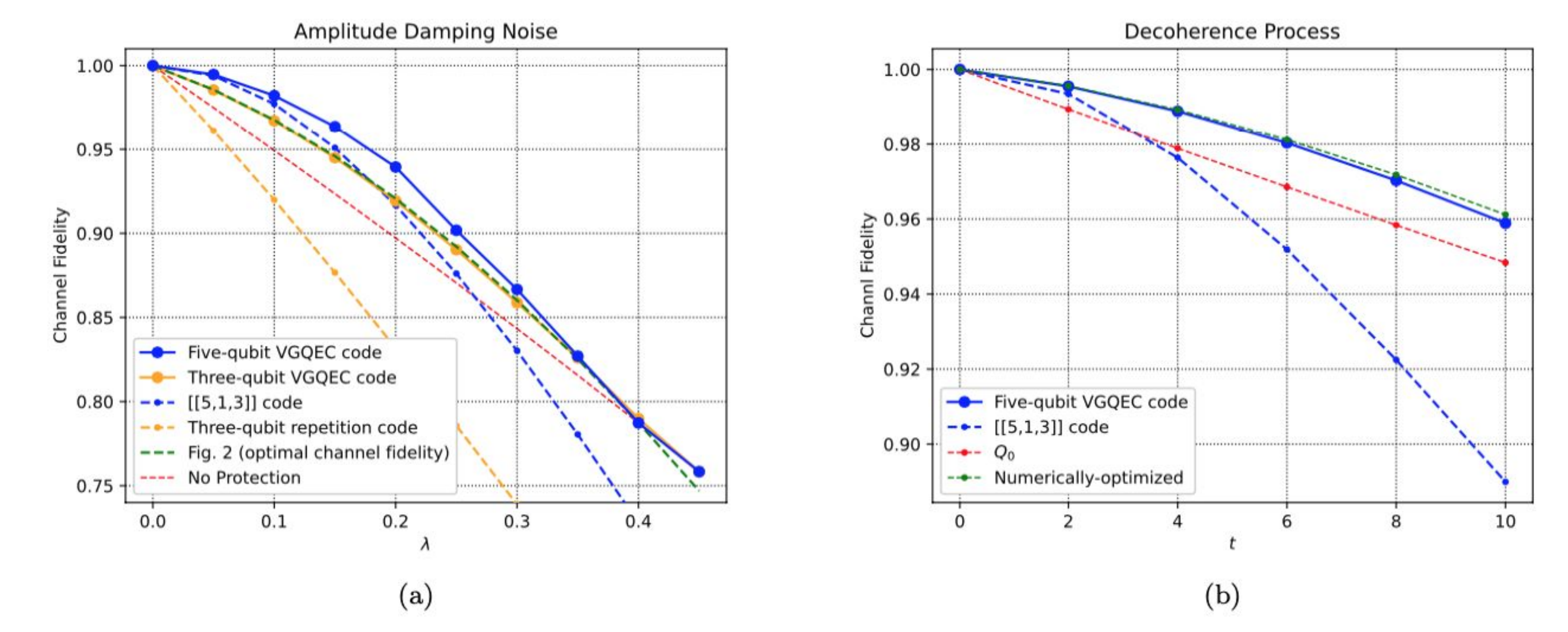
$$\mathcal{R}_{out} = \mathcal{R}_c \circ U_{\mathcal{R}}(\vec{\beta}_{opt}).$$

VGQEC



Simulation-based optimization results (theoretical exploration):
a) Three-qubit VGQEC code optimized to handle amplitude damping noise.
b) Five-qubit VGQEC code is tested under changing noise conditions over time.

VGQEC

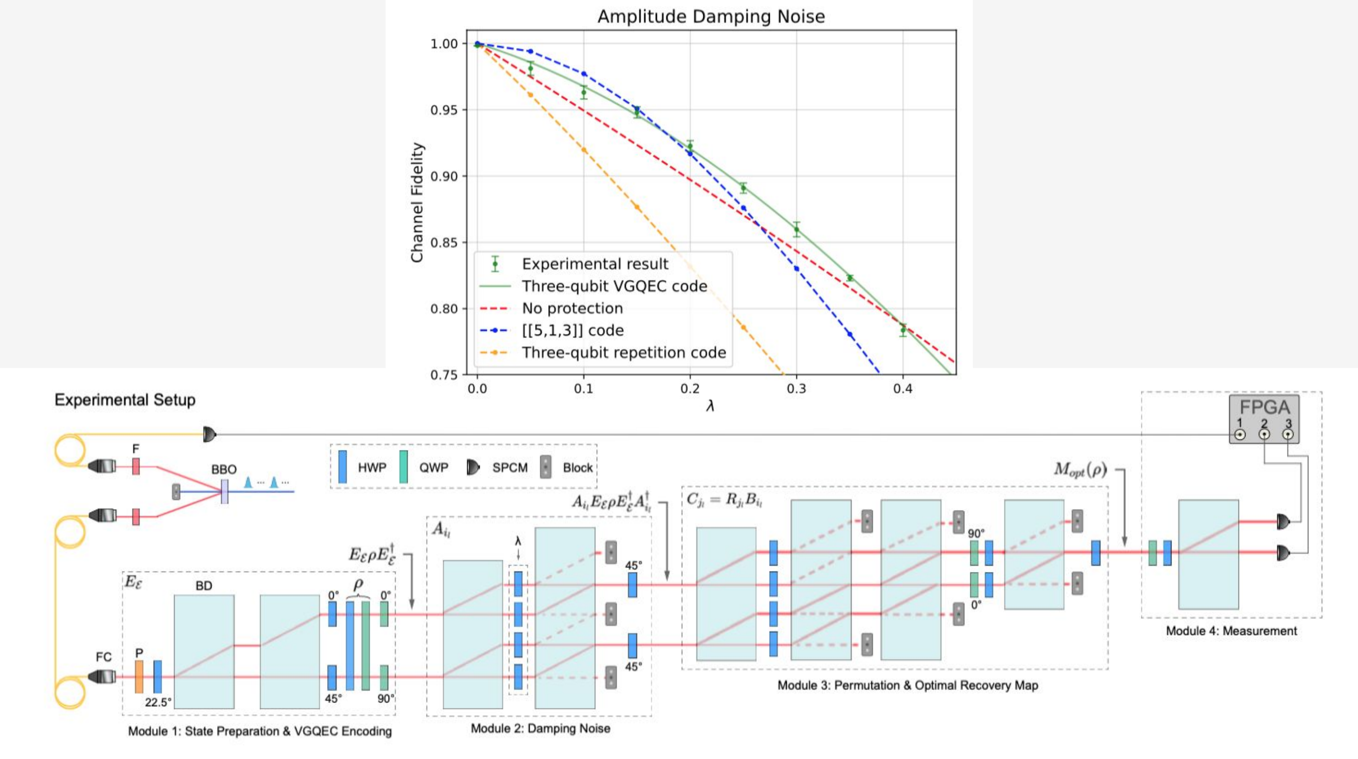


(a)

(b)

Circuit-level implementation and performance (practical realization):
a) Three-qubit VGQEC code's fidelity compared under amplitude damping noise.
b) Five-qubit VGQEC code evaluated under IBM-LIMA machine data, uneven noise thermal relaxation.

VGQEC



Experimental setup: Uses photons and optical elements to physically implement and test three-qubit VGQEC code.

VGQEC Limitations

1. Parameter Placement Unclear
2. Recovery is Challenging
3. Encoding Not Yet Optimized

```
-function(a){return!this.B[a.getId()]};_.k.
p(this)&&ip(this).Ud()};_.k.ti=function(a){this
o[a].bd(!0))};_.k.Vd=function(a){this.o[a.getId()
on(a,c){this.o.push({Jc:a,options:c})};
dow.gapi={};var e=window.___jsl={};e.h=_.J(_.F(a,
&&(d=_.F(d,2))&&this.b.push(d);_.x("gapi.load",(0
s.C=a;this.w=this.b=null;this.D=0;this.B=
ORD("******");0<=a.indexOf("MSIE")&&0<=a.indexOf(
arseFloat)(a[1])&&(this.o=!0)};_.z(kp,_.A);
nstanceof Array)for(var e in d)lp(a,c,d[e]);else{
er?c.addEventListener(d,e,!1):c&&c.attachEvent?c.a
o)return null;if(c instanceof Array){var d=null,e
.w==a&&(d=this.b,this.b=null);if(e=a.getAttribute(
stener(c,e,!1):a.detachEvent&&a.detachEvent("on"+c
;this.w=a;c.preventDefault?c.preventDefault():c.re
gp(window,_.J(_.F(c,8)));c=_.ec();var d=_.W();a=ne
nction(a){try{a()}catch(g){d.log(g)}},this));_.yi(
Y.PASSWORD("******"),"DOMContentLoaded"); cp(windo
;_.x("gbar.mls",function(){});_.Ma("eq",new kp(_.W
unction(){for(var a=function(a){return function(){
.Ia.U();_.Ja(e,"api").Ra();fp(_.Ja(e,"m"),functio
c&&(c=c.querySelector(".gb_b"))&&lp(_.yi("eq")
op=/(\s+|^)gb_2f(\s+|$)/;np&&!op.test(np
```

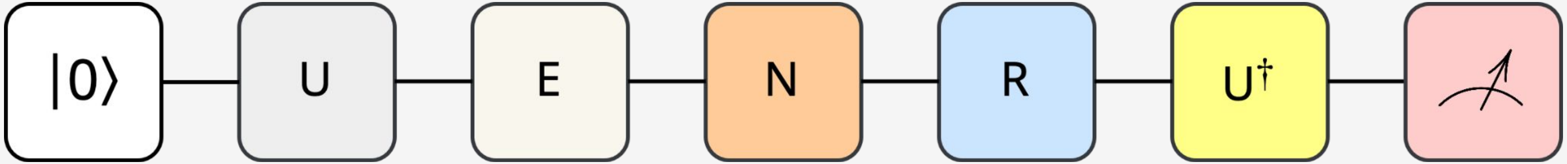
III.

Implementation

Scope of Implementation

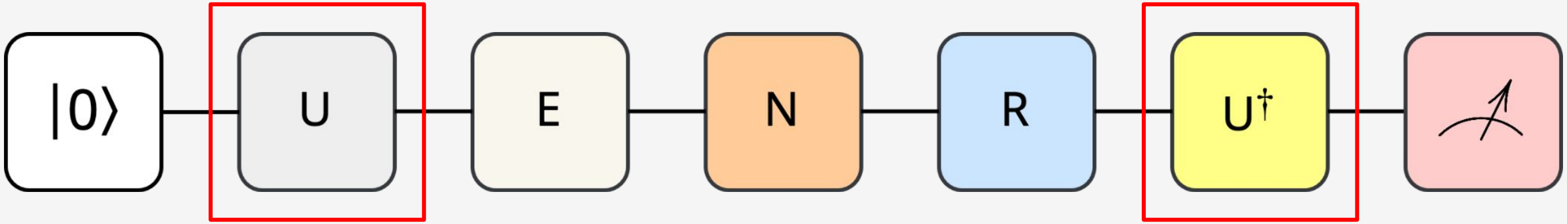
1. No quon language/graph
2. No SDP for optimal recovery map
3. Less methods for comparison
4. No photonic realization
5. Did not implement optimizer

Implementation



Implemented Quantum Circuit

Implementation

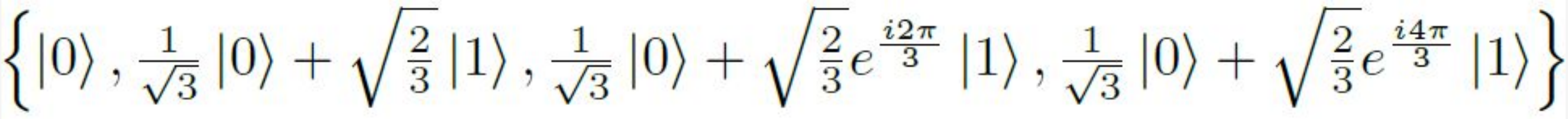


Unitary

Implementation

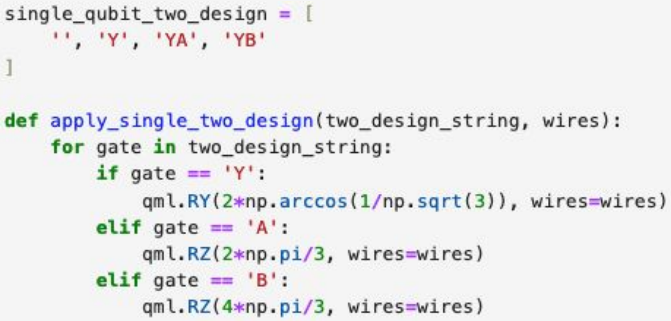
Unitary

From paper



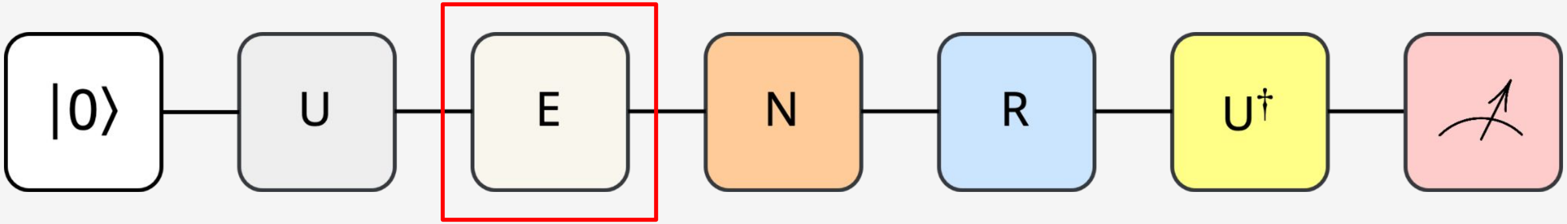
$$\left\{ |0\rangle , \frac{1}{\sqrt{3}} |0\rangle + \sqrt{\frac{2}{3}} |1\rangle , \frac{1}{\sqrt{3}} |0\rangle + \sqrt{\frac{2}{3}} e^{\frac{i2\pi}{3}} |1\rangle , \frac{1}{\sqrt{3}} |0\rangle + \sqrt{\frac{2}{3}} e^{\frac{i4\pi}{3}} |1\rangle \right\}$$

Our implementation



```
single_qubit_two_design = [
    '', 'Y', 'YA', 'YB'
]

def apply_single_two_design(two_design_string, wires):
    for gate in two_design_string:
        if gate == 'Y':
            qml.RY(2*np.arccos(1/np.sqrt(3)), wires=wires)
        elif gate == 'A':
            qml.RZ(2*np.pi/3, wires=wires)
        elif gate == 'B':
            qml.RZ(4*np.pi/3, wires=wires)
```

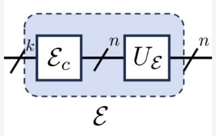
Implementation



Encoding

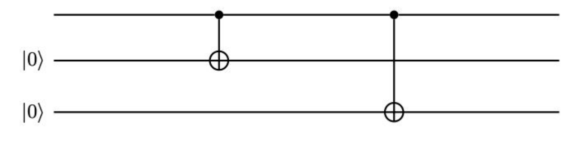
Implementation

Encoding - 3 qubit VGQEC

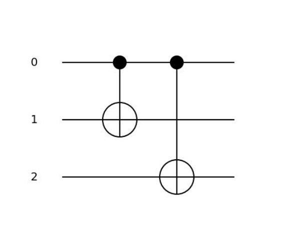


Original

From paper

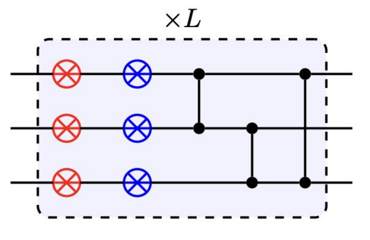


Our implementation

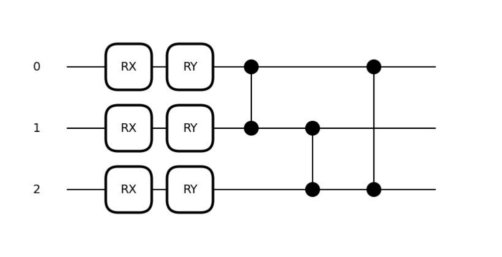


VQC

From paper

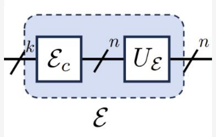


Our implementation



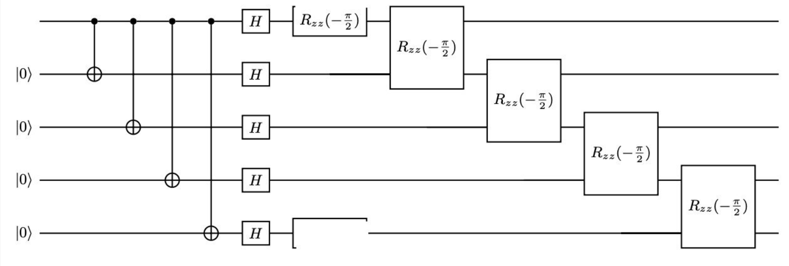
Implementation

Encoding - 5 qubit VGQEC



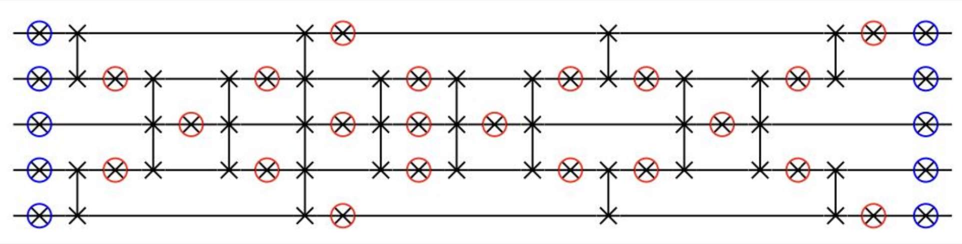
Original

From paper

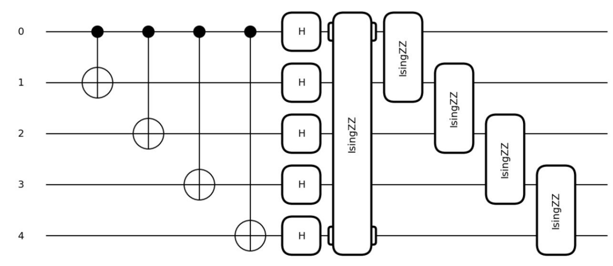


VQC

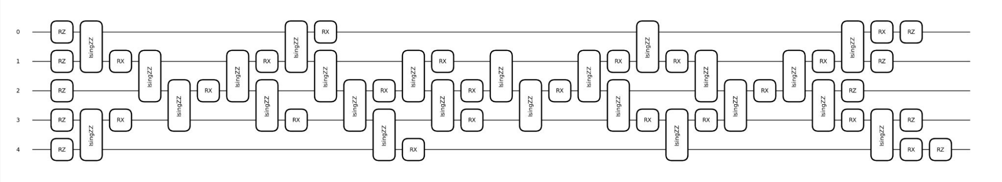
From paper



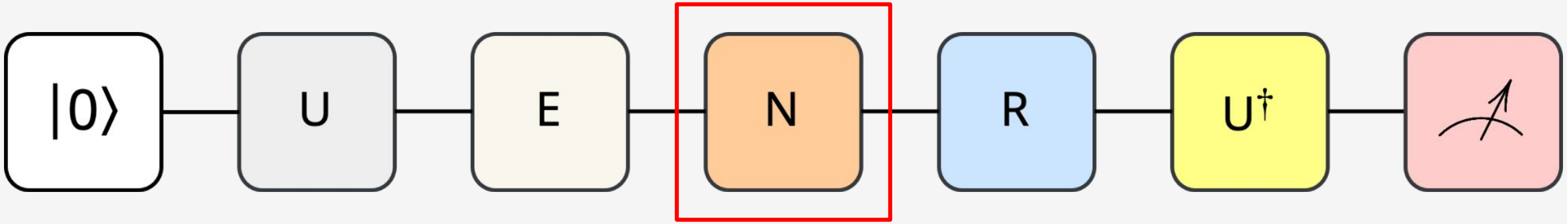
Our implementation



Our implementation



Implementation



Noise

Noise - Amplitude Damping

From paper

$$\mathcal{N}_s^{ad}(\rho) = \sum_{k=0,1} E_k \rho E_k^\dagger,$$

$$E_0 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\lambda} \end{bmatrix} \quad E_1 = \begin{bmatrix} 0 & \sqrt{\lambda} \\ 0 & 0 \end{bmatrix}$$

Our implementation

```
qml.AmplitudeDamping(gamma, wires=wire)
```

Implementation

Noise - Thermal Relaxation

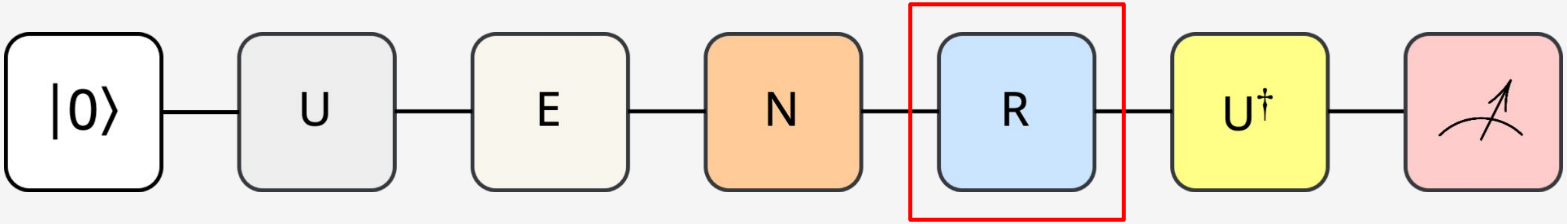
From paper

$$A_1 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma-\lambda} \end{bmatrix} A_2 = \begin{bmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{bmatrix} A_3 = \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{\lambda} \end{bmatrix}$$

Our implementation

```
qml.ThermalRelaxationError(0, T1, T2, t, wires=wire)
```

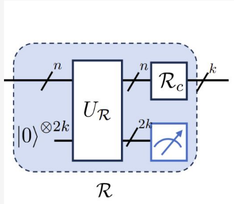
Implementation



Recovery

Implementation

Recovery- 3 qubit VGQEC



Original

From paper

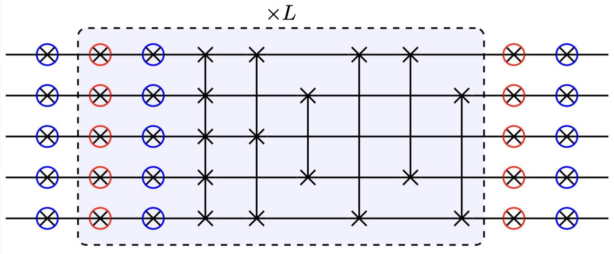
Original recovery

Our implementation

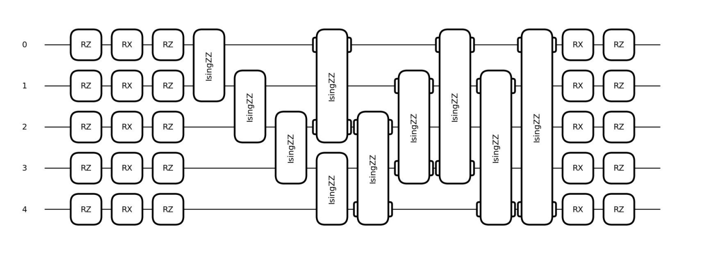
Not needed (assumption)

VQC

From paper

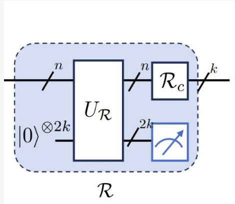


Our implementation



Implementation

Recovery- 5 qubit VGQEC

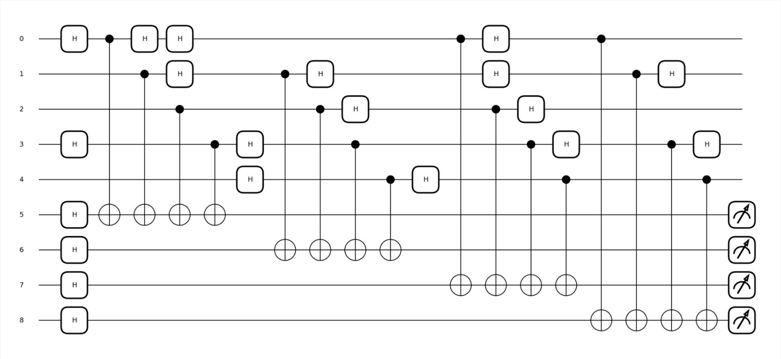


Original

From paper

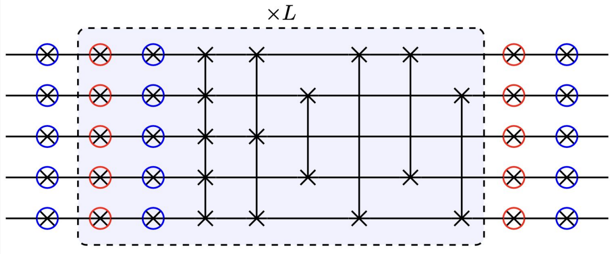
Original recovery

Our implementation

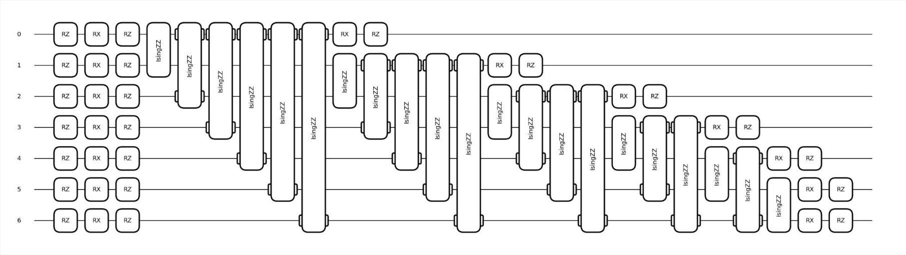


VQC

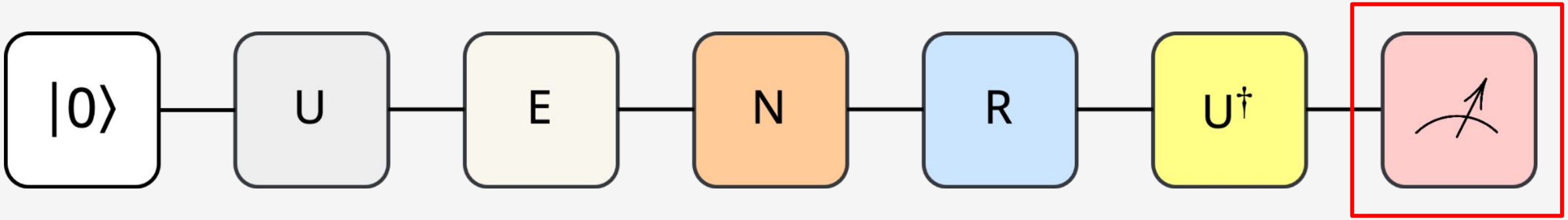
From paper



Our implementation



Implementation

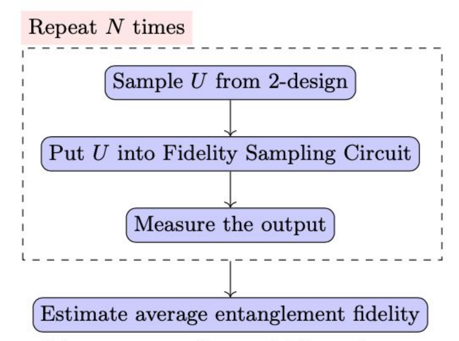


Fidelity

Implementation

Fidelity

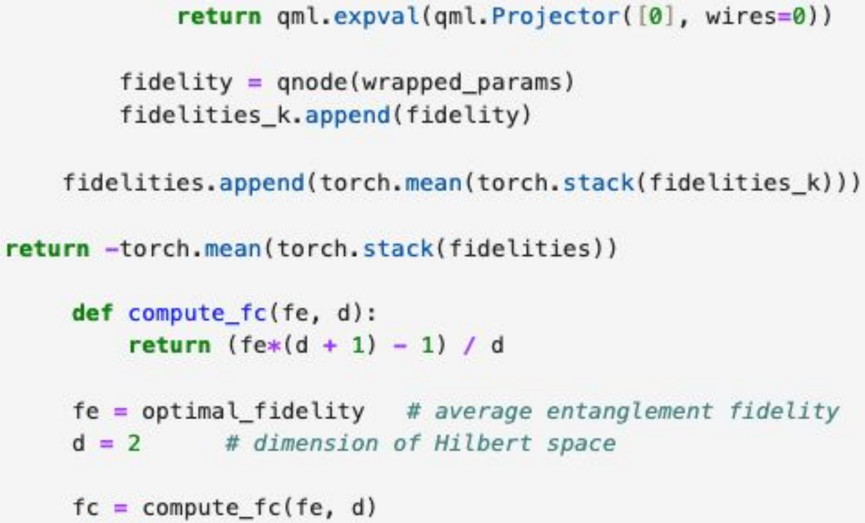
From paper



Repeat N times
Sample U from 2-design
Put U into Fidelity Sampling Circuit
Measure the output
Estimate average entanglement fidelity

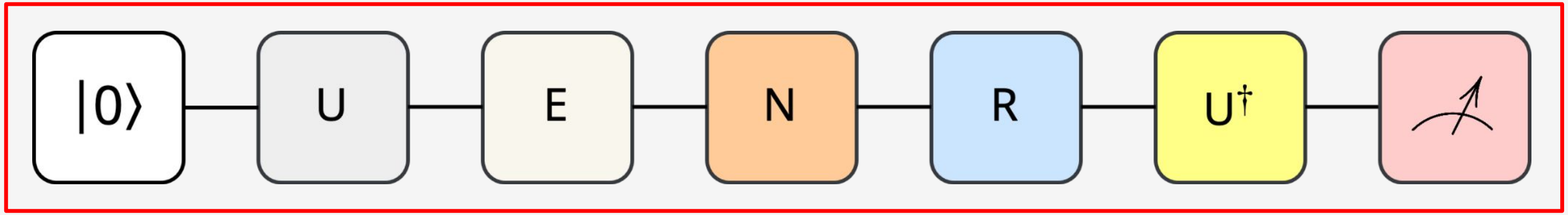
$$\overline{F}_e(\mathcal{M}) = \frac{dF_C(\mathcal{M})+1}{d+1}.$$

Our implementation



```
            return qml.expval(qml.Projector([0], wires=0))

        fidelity = qnode(wrapped_params)
        fidelities_k.append(fidelity)

    fidelities.append(torch.mean(torch.stack(fidelities_k)))

return -torch.mean(torch.stack(fidelities))

def compute_fc(fe, d):
    return (fe*(d + 1) - 1) / d

fe = optimal_fidelity   # average entanglement fidelity
d = 2      # dimension of Hilbert space

fc = compute_fc(fe, d)
```

Implementation



$|0\rangle$ U E N R $U^\dagger$

Optimization

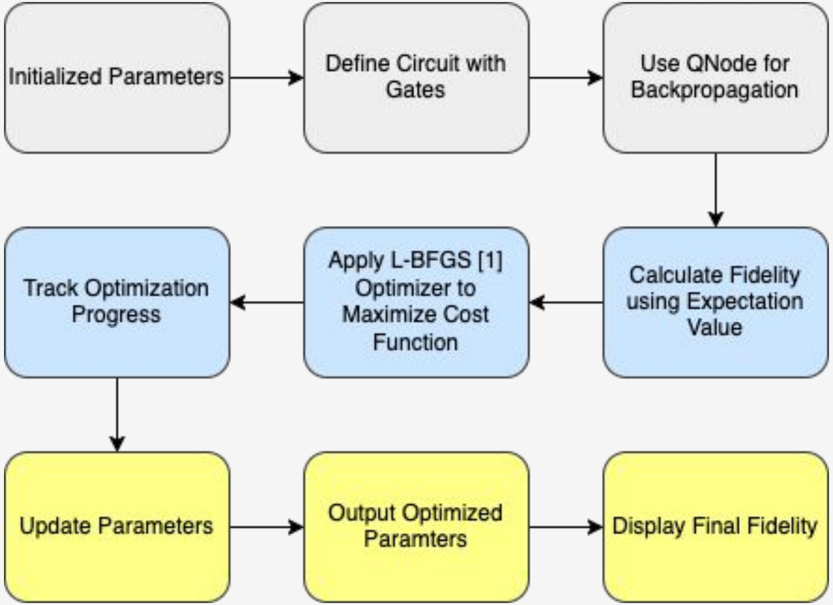
Implementation

Optimization

From paper

- D. C. Liu and J. Nocedal, “On the limited memory bfgs method for large scale optimization,” Mathematical pro- gramming, vol. 45, no. 1, pp. 503–528, 1989.
- Does not specify how many parameters they are tuning

Our implementation



Initialized Parameters

Define Circuit with Gates

Use QNode for Backpropagation

Calculate Fidelity using Expectation Value

Apply L-BFGS [1] Optimizer to Maximize Cost Function

Track Optimization Progress

Update Parameters

Output Optimized Paramters

Display Final Fidelity

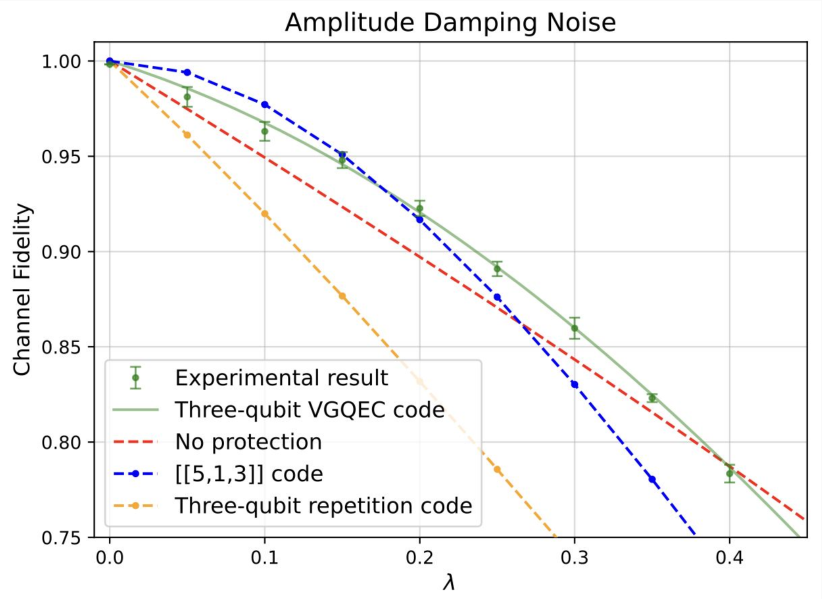
[1] https://pytorch.org/docs/stable/generated/torch.optim.LBFGS.html

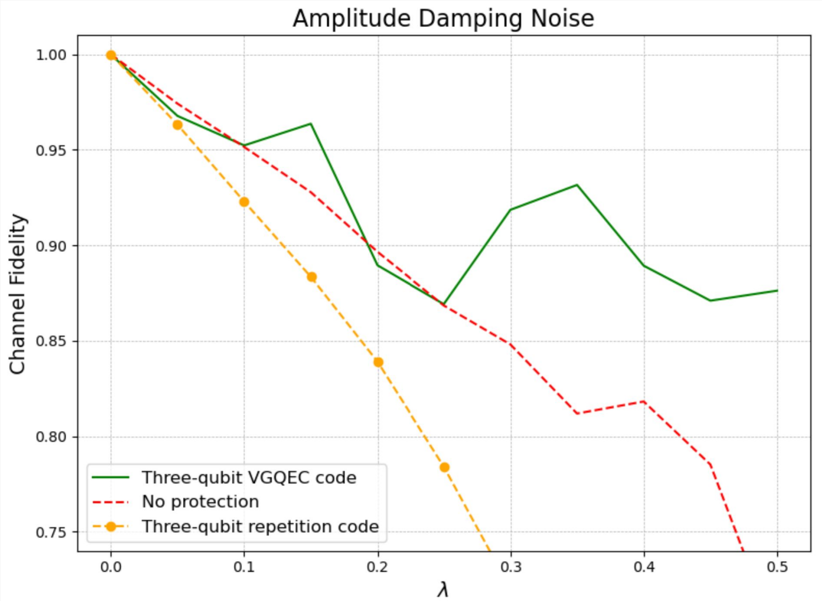
IV.

Results

Results

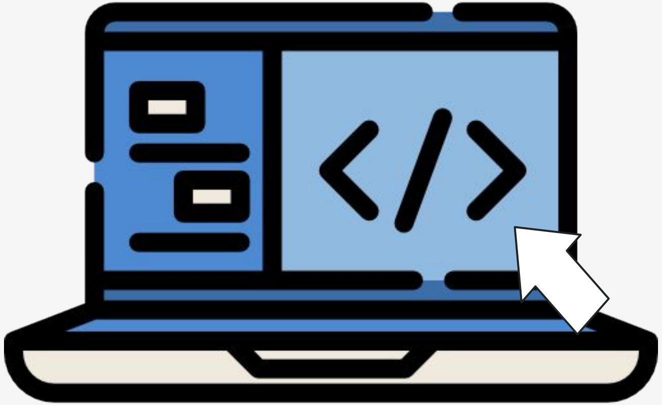
Simulated Graphs





Results

Demo



Thank you!

Questions?

Resoruces:

https://research.google/blog/making-quantum-error-correction-work/?utm_source=chatgpt.com

http://theory.caltech.edu/~preskill/ph229/notes/chap7.pdf

https://www.dhuality.com/posts/2022-09-27-qec513code/#:~:text=In%20this%20code%2C%20five%20physical,psi%5Crangle%20%E2%88%A3%CF%88%E2%9F%A9.