QBadgers

Alice & Bob Challenge

ALICE & BOB

sparsh
srivastava

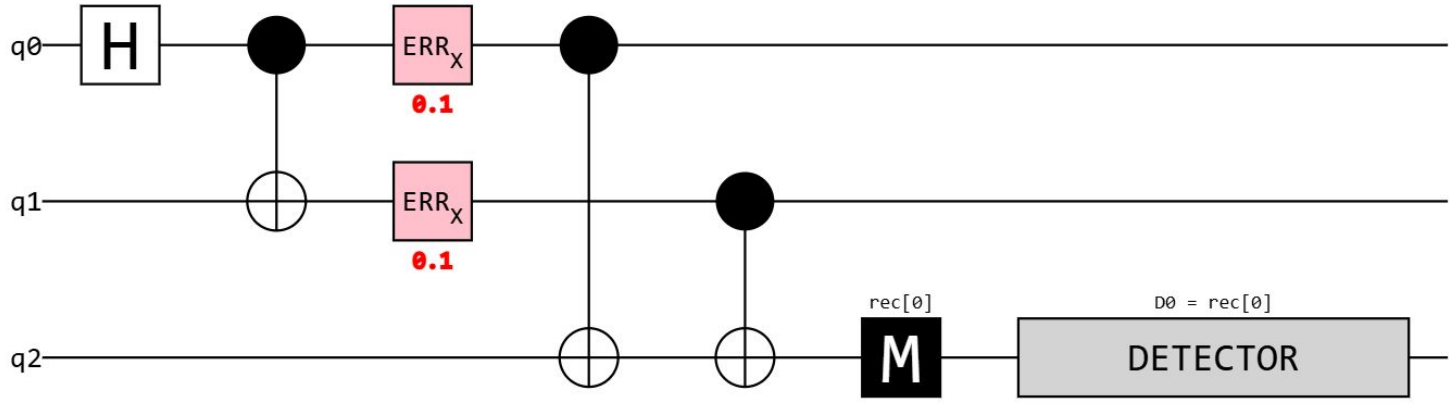
kiplimo
kemei

henry
lin

akshat
prakash

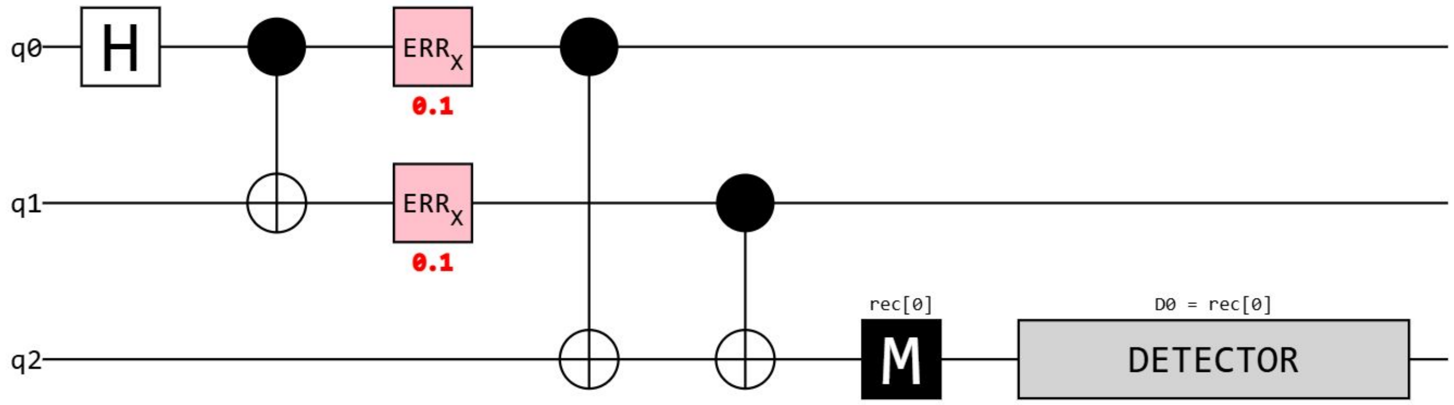
ruben
aguilo schuurs

Task 2: bell state



demonstrates error detection without destructive measurement of data qubits

Task 2: bell state



1 - p*(1-p)*2

with X errors on both at p=0.1

- **0.82067%** rate of success (simulated 0.82%)

Task 3: repetition code decoder

Logical Qubits: $|0\rangle_L = |000...0>$ $|1\rangle_L = |111...1>$

Stabilizer group: Parity check $Z_i Z_{i+1}, \quad i = 0, \ldots, n-2$

Example:

Error!

Data: $0\ 0\ 0\ 0\ \boxed{1}$

Syndrome: $0\ 0\ 0\ \boxed{1}$

Syndrome 0: No error or both error
Syndrome 1: one error

three decoding algorithms

1. Majority wins

- Data: more 0's than 1's? Likely 0!

d: 0 0 0 0 1
s: 0 0 0 1

👌

2. "Smart" majority *(or so we thought)*

- 0 in syndrome → no error or error in 2 qubits
  - More likely no error!
  - Keep track of pairs coupled to syndrome, and majority from there

d: 0 0 0 0 1
s: 0 0 0 1

👌

three decoding algorithms

$$1\,0\,0\,0\,1$$

$$1\,0\,0\,1$$

Majority works! :)

$$1\,1\,0\,0\,1$$

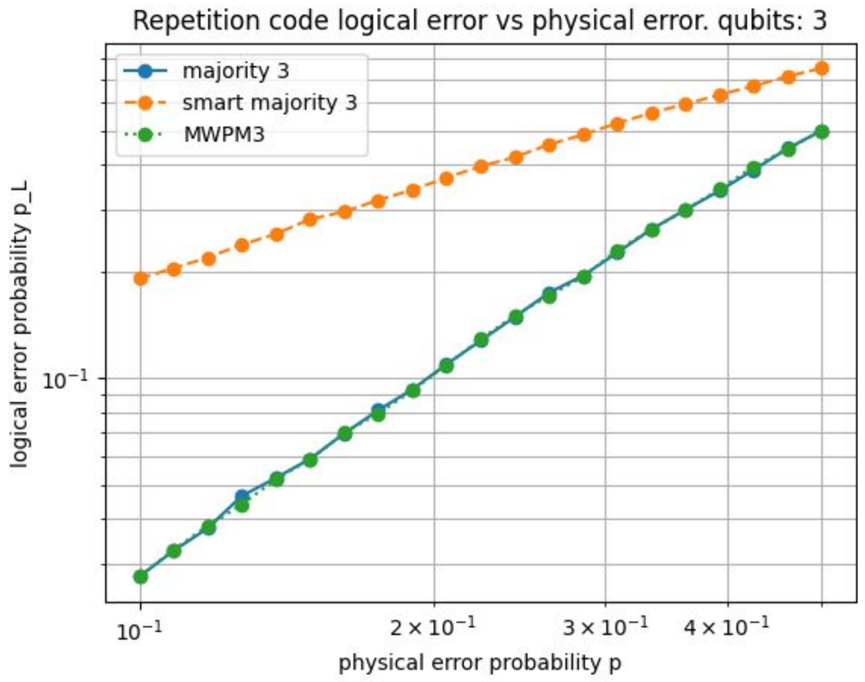
$$0\,1\,0\,1$$

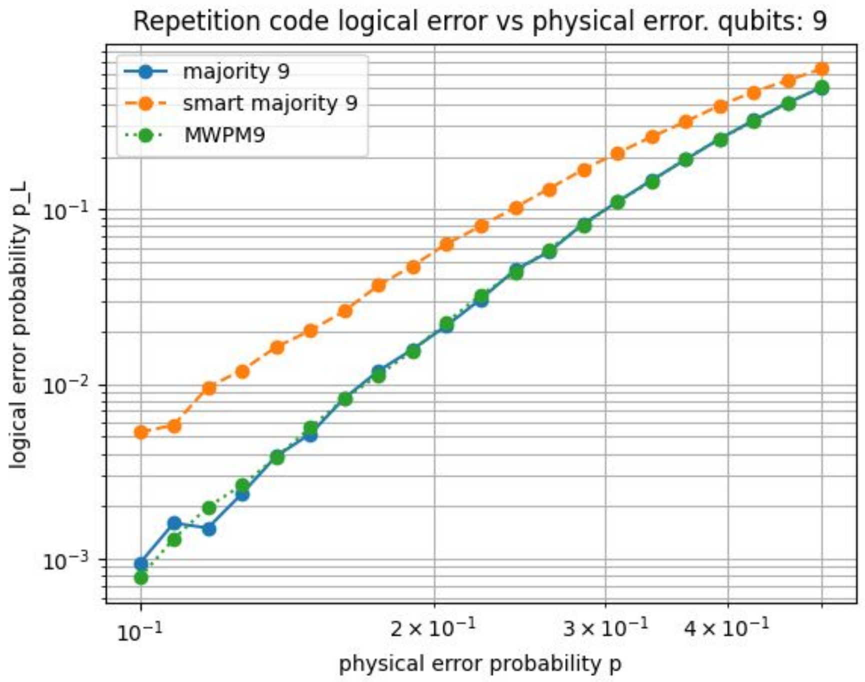
Error correction limit, any decode fails!

$$\left\lfloor \frac{n-1}{2} \right\rfloor$$

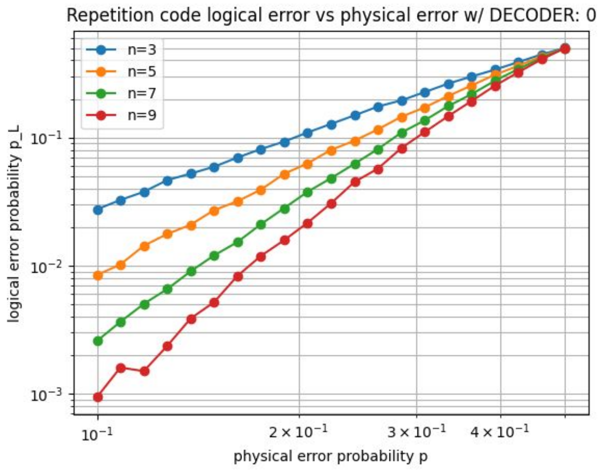
Finding: For repetition code decoding, “**majority wins” is already optimal!**

majority vs smart majority vs minimum-weight perfect matching (MWPM)

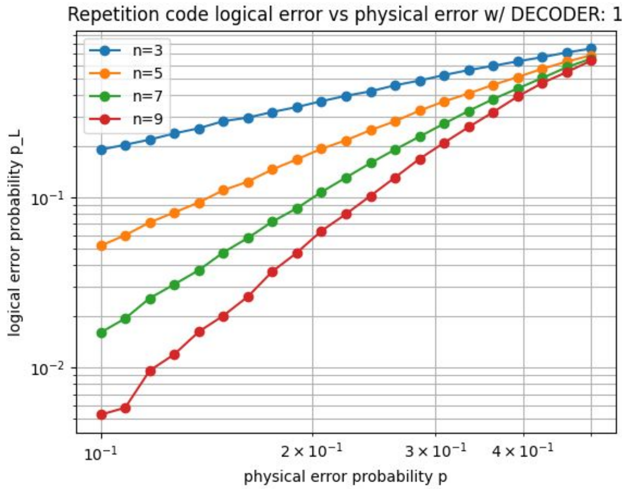




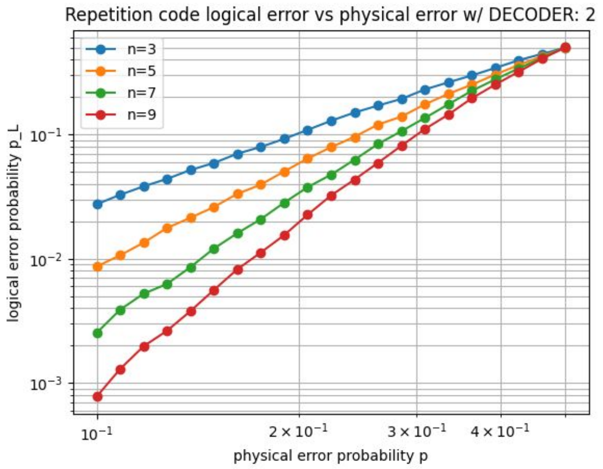
majority



smart majority



MWPM



Task 4.1: Cat-Repetition Codes Exploiting Noise Bias

Spiel

- Cat qubits exponentially suppress phase-flip (Z) errors
- Residual noise dominated by bit-flip (X) errors
- Enables repetition codes to operate near their optimal regime

Setup

• Physical bit-flip probability: $p_x = 0.01$

• Noise bias ≈ $10^8 \Rightarrow$ p_z ≈ 10^{-10}

• Code distance d = number of data qubits

Task 4.1: Cat-Repetition Codes Exploiting Noise Bias

Key Observation

• Logical error rate decays exponentially with distance
• Below threshold: log(p_L) ∝ −d

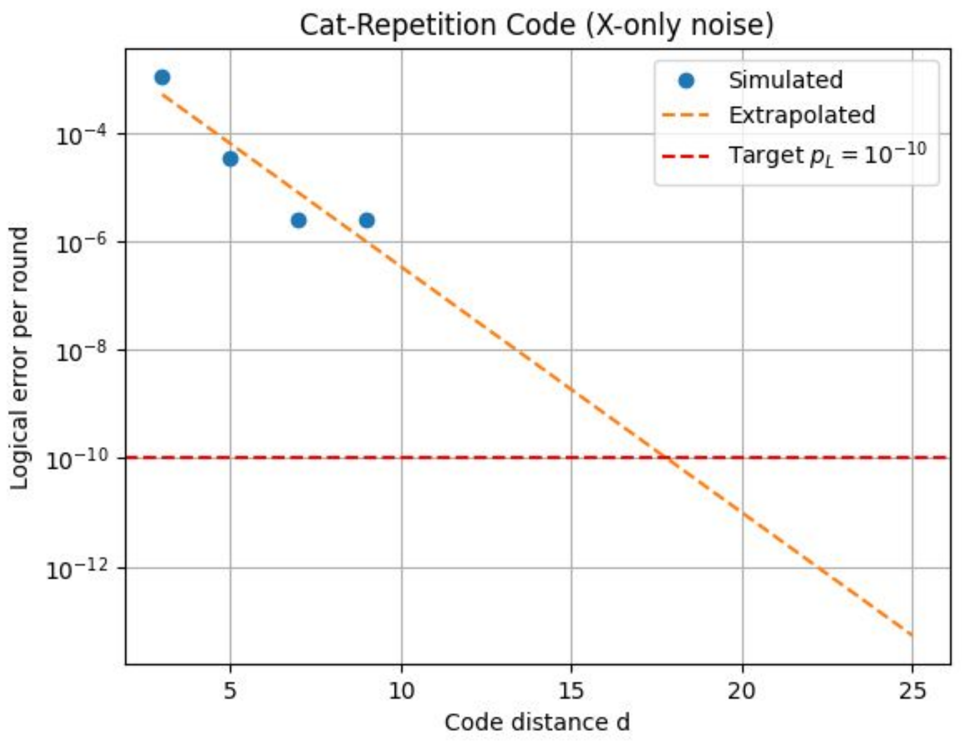
Result

• Extrapolation predicts d ≈ 18–20 for p_L ≈ 10^{-10}

Takeaway

• Cat-repetition codes achieve very low logical error rates

• But incur linear qubit overhead per logical qubit



(1D repetition code)

Task 4.2: Surface Code Benchmarks & Comparison

Surface code protects against both X and Z errors

Evaluated under full circuit-level depolarizing noise (after each Clifford)

Physical error rate fixed at $p = 10^{-3}$

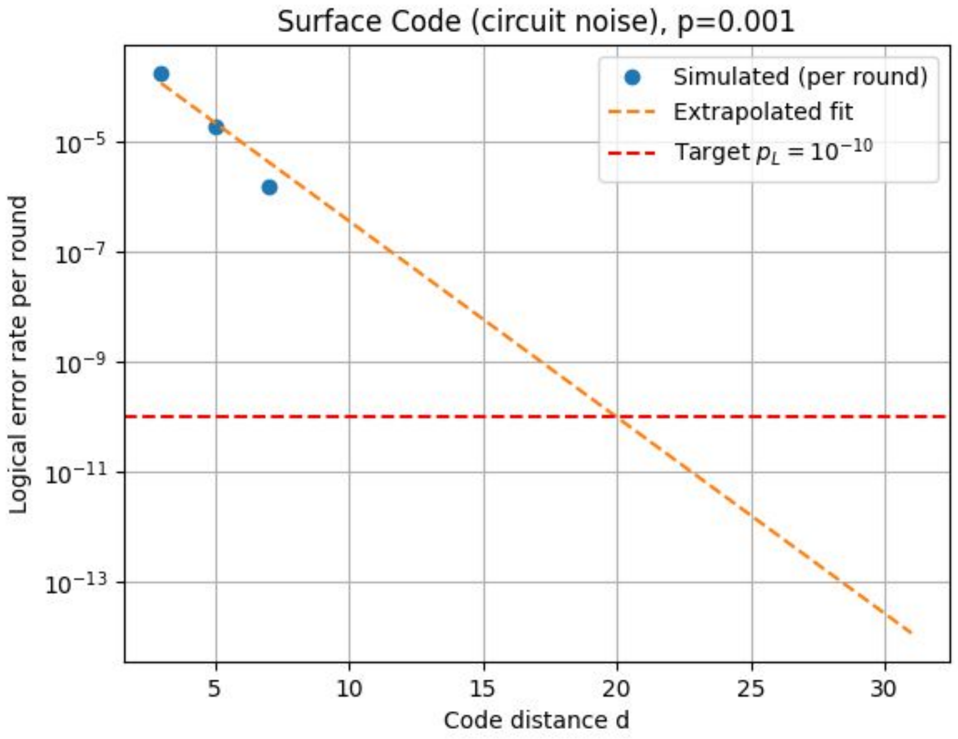
Key Results

• Logical error rate decreases exponentially with distance

• Threshold behavior observed

• Extrapolation gives required distance d ≈ 20 to reach p_L ≈ 10^{-10}

• Corresponds to ≈ 400 data qubits (d^2 scaling)



Task 4.2: Cat-Repetition vs Surface Code

Cat-Repetition

- 1D structure
- Linear qubit overhead
- Extremely effective under strong noise bias

Surface Code

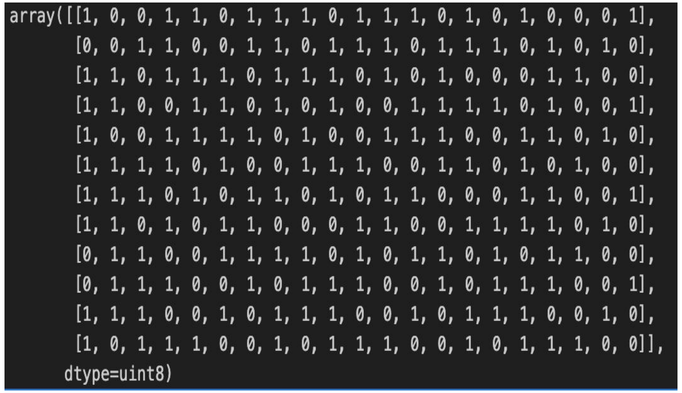
- 2D structure
- Quadratic qubit overhead
- Works under general (unbiased) noise

Key Insight

• Exploiting noise bias can reduce fault-tolerant overhead by orders of magnitude

• Motivates LDPC-cat architectures combining high rate, locality, and bias awareness

Task 5: Reed-Solomon Code

H = 

```
array([[1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1],
       [0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0],
       [1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0],
       [1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1],
       [1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0],
       [1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0],
       [1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1],
       [1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0],
       [0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0],
       [0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1],
       [1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0],
       [1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0]],
      dtype=uint8)
```

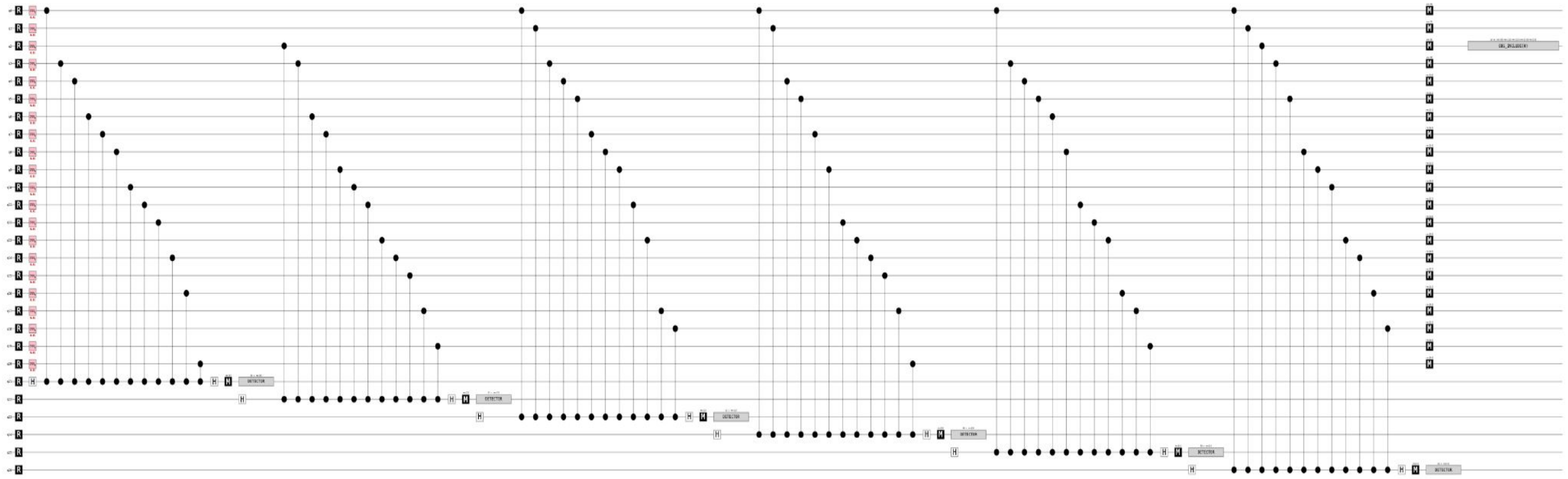
[n, k, d] = [7, 5, 3]
n = symbols of codeword
k = # logical qubits
d = code distance, detect d-1, correct (d-1)/2

The method from Notebook 2 generalizes to any classical code with a parity check matrix (generally CSS Codes)

1. Construct the parity check matrix by using the Galois package.
2. Each row corresponds to one parity check. One corresponds to where the Z-stabilizer is going to check.
3. The circuit is constructed using the same method from the notebook, CX gates from stabilizer qubit to ancilla (per row)

Quantum Circuit

[7, 5, 3]

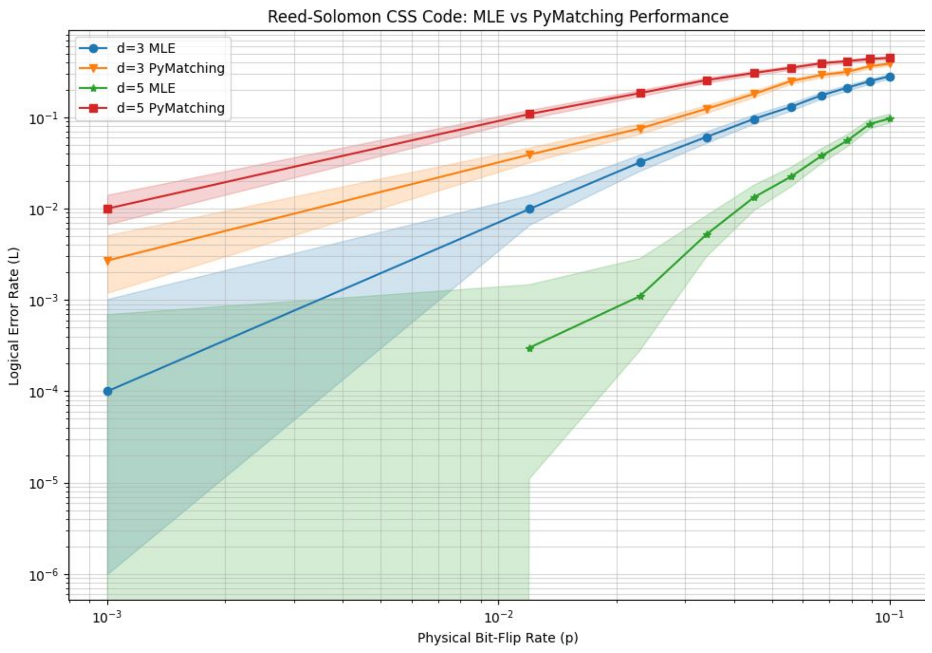


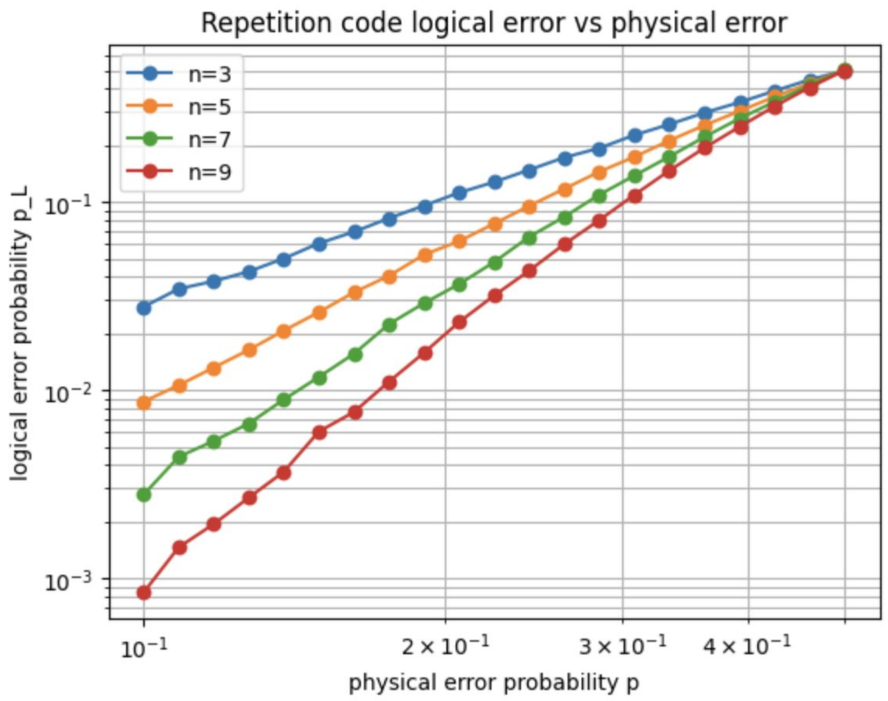
Decoding Strategies

1. Maximum Likelihood Estimation
    a. Brute force optimizer: it searches for the most statistically likely logical state given the syndrome. It looks at every possible error e satisfying H*e = syndrome and picks the most probable error
2. Minimum weight perfect matching (MWPM) - pymatching
    a. MWPM is designed for surface codes and not well suited for dense codes like this. It does not work for all parameters of the Reed-Solomon code.

We tried these two decoding methods and compared their performances.

Results

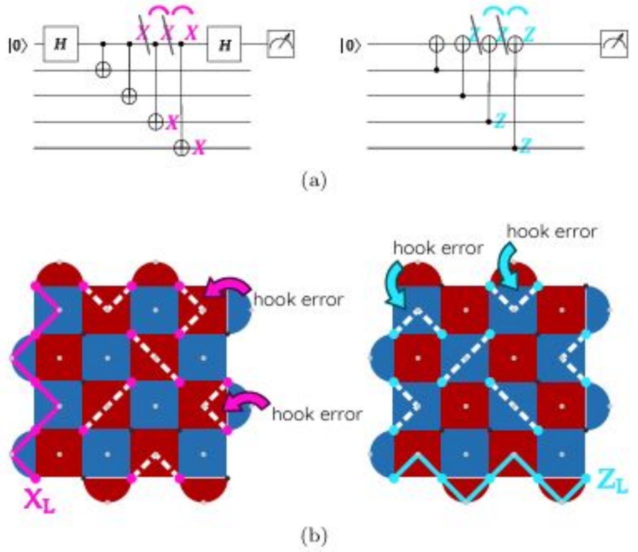




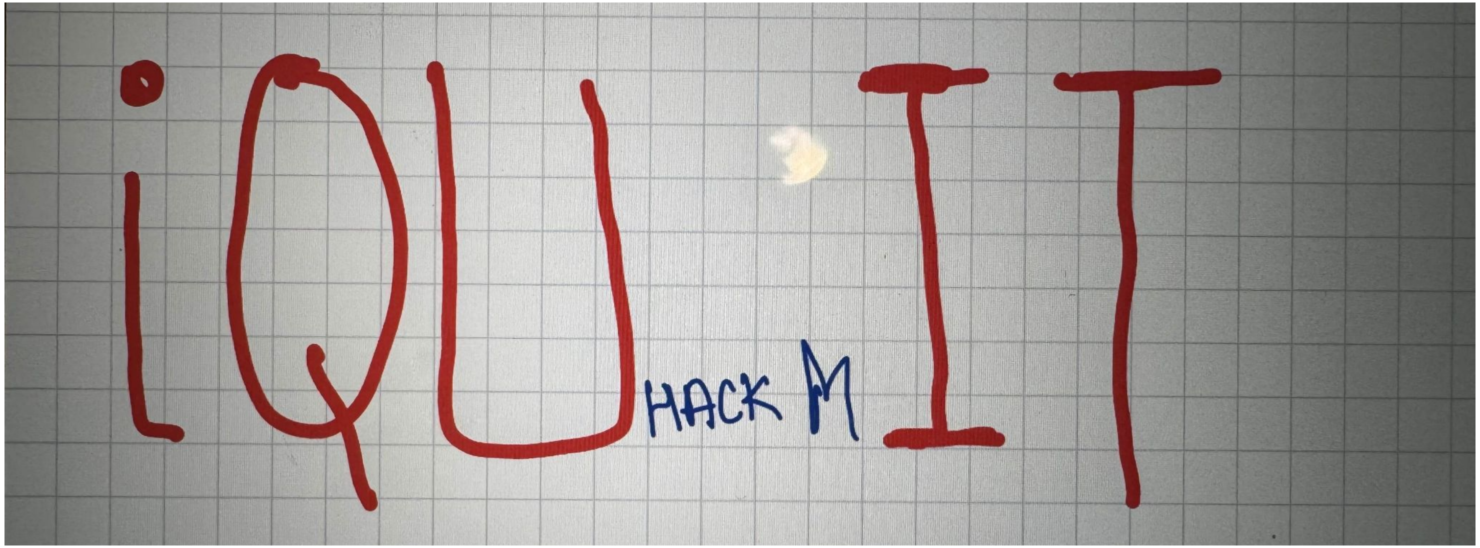
The threshold is not reasonable to consider using this code in practice.

Next Steps: Further Considerations

- The Reed-Solomon code requires high-weight (10 stabilizers) stabilizers, so this code will be highly susceptible to error propagation like hook errors on the surface code.
- Surface code hook errors can be mitigated by reordering the syndrome extraction circuits to propagate errors against the grain.
- Not only does this code have the chance to propagate 10 errors, but it's also not clear if these errors can be mitigated.



Hook errors on surface code



THANK YOU!