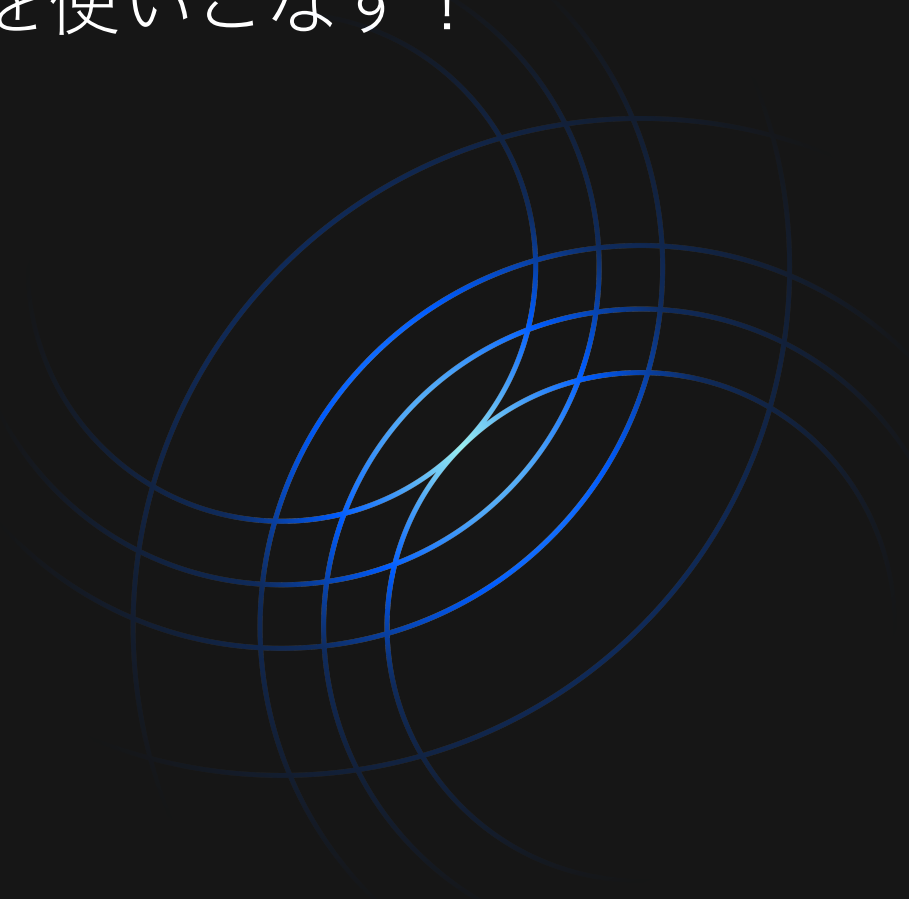
Qiskit Runtime Primitivesを使いこなす！

Ikko Hamamura
IBM Quantum, IBM Research - Tokyo



自己紹介

- 名前：濵村 一航
- Research scientist, Quantum
- Doctor of Engineering at Kyoto University (March 2020)
- Join IBM Research – Tokyo in 2020
- Qiskit Advocate
- Code owner of Qiskit Terra.
- Member of primitives.

- アカウント名: ikkoham で活動してます
- 教科書の翻訳をしました 「量子コンピュータシステム」

このトークの目的

IBM **Quantum**

– Primitivesに関する概念・言葉に出会うこと
– 帰ってチュートリアルをやってみようと思ってもらうこと
– 実機を身近に感じてもらうこと

話さないこと
– 豊富なコード例
– 数学的・物理的・実装的な詳細

目次

1.Primitivesとは？

2.Samplerで確率分布のサンプリングをする

3.Estimatorで期待値を推定する

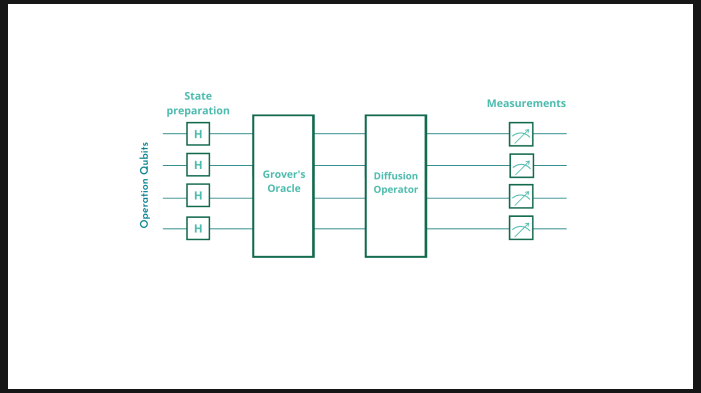
4.量子コンピュータのエラーを緩和する

目次

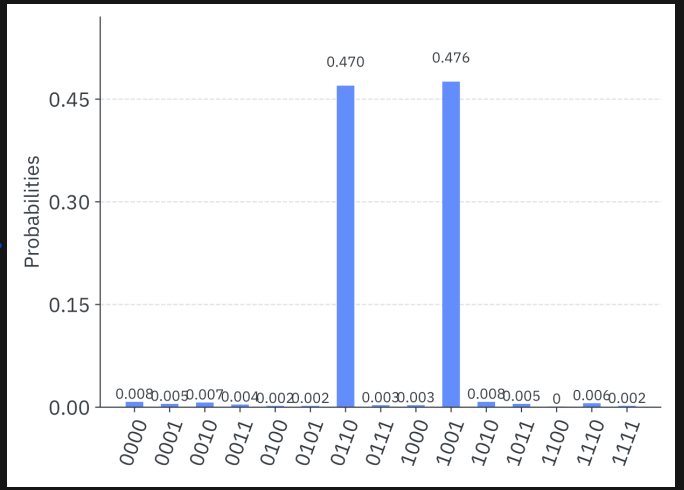
What is necessary for algorithms?

IBM **Quantum**

Recap Grover's algorithm



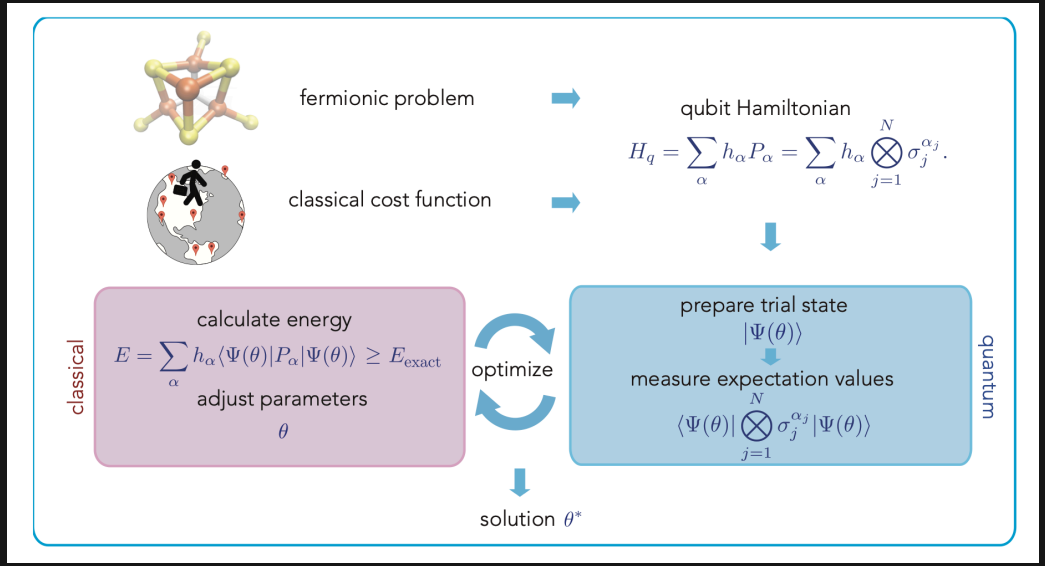
Quantum Circuit



Probability distribution

What is necessary for algorithms?

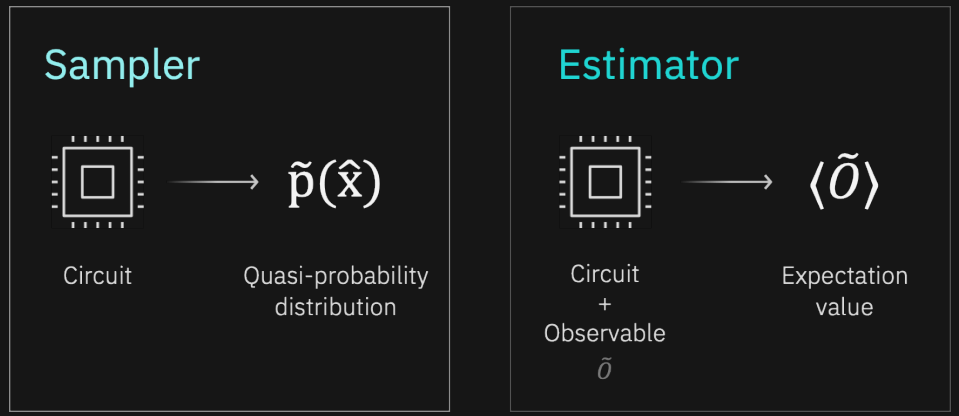
Recap Variational Quantum Eigensolver



[1] Alberto Peruzzo, et al., A variational eigenvalue solver on a photonic quantum processor, Nat. Commun. **5** (2014), 4213.

Requirements from algorithms

- Sample probabilities from quantum circuits $P(x) = |\langle x|\psi\rangle|^2 = |\langle x|U|0\rangle|^2$
  - Grover, Amplitude amplification, Quantum phase estimation…
- Estimate expectation values of quantum observables $\langle\psi|O|\psi\rangle = \langle 0|U^\dagger OU|0\rangle$
  - VQE (Variational Quantum Eigensolver), QAOA (Quantum Approximate Optimization Algorithm)

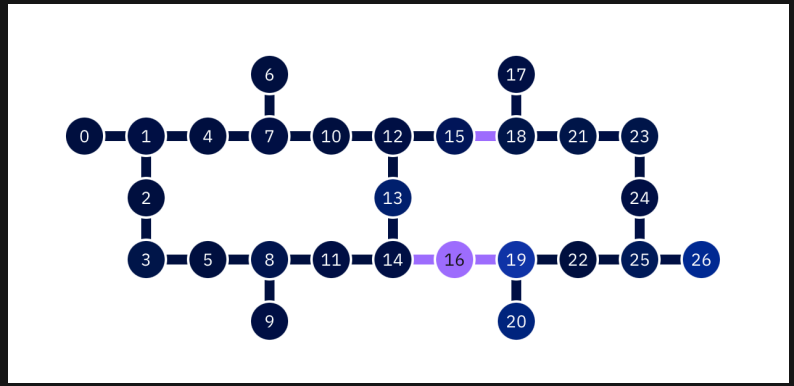


Can the quantum devices run any circuit?

The answer is NO!

- Basis gates
  - The gates supported by the quantum computers are limited.
- Coupling map
  - Not all qubits are connected.

→ Transpiler is necessary.



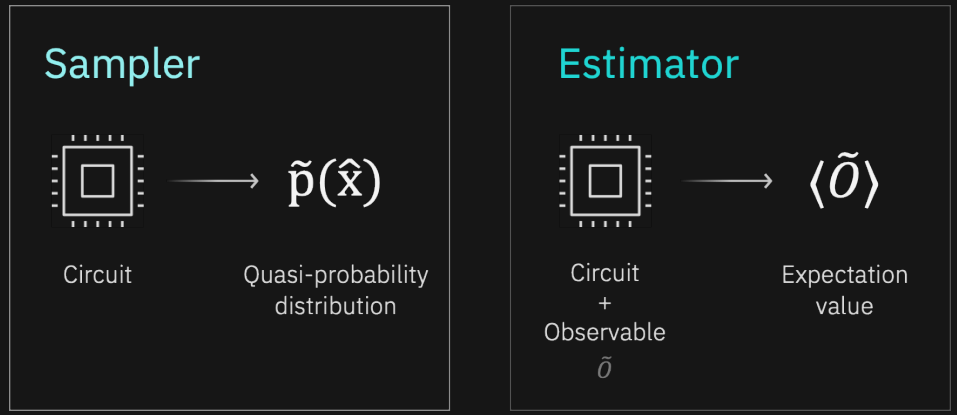
Difficult to use a noisy quantum computer directly

- Need to transpile to satisfy the constraints of quantum computers
- Need to mitigate and suppress errors

→ Need abstraction for algorithm and application developers/researchers!

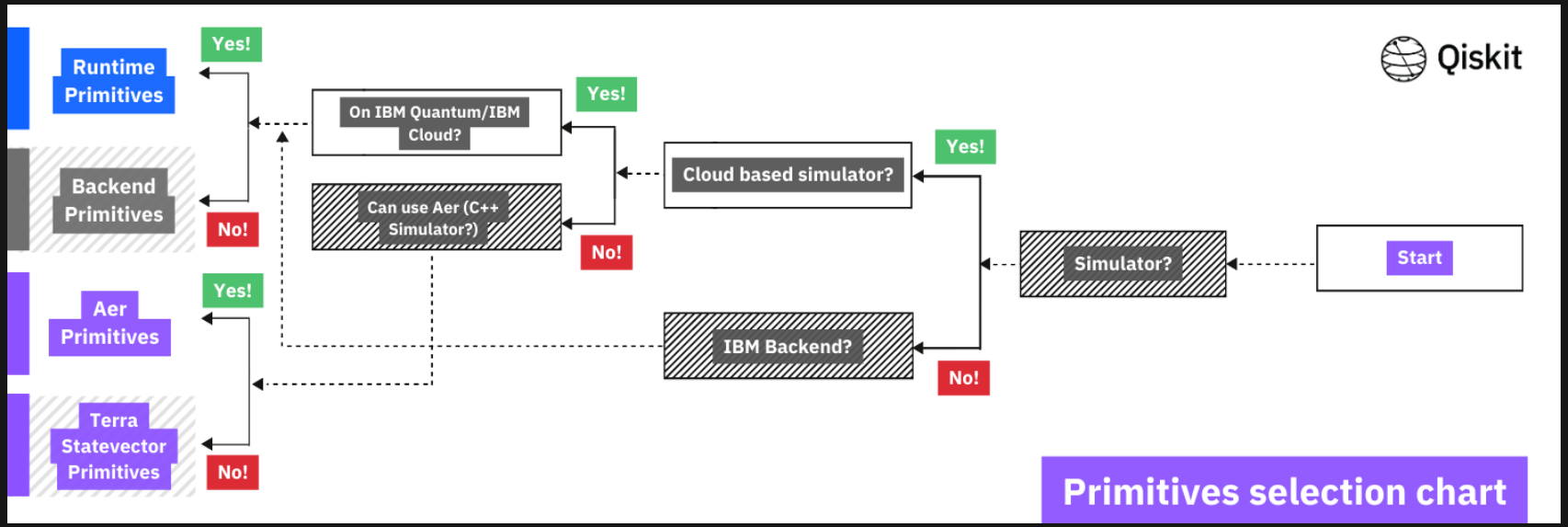
Abstraction of backend

– Algorithm developers do not want to get into the details of actual devices and usage.
– Need abstraction limited to APIs needed for algorithms.
– This simple interface can provide transpile and error mitigation inside the APIs.
– Two primitives: Sampler and Estimator



Various IBM's primitives

IBM **Quantum**

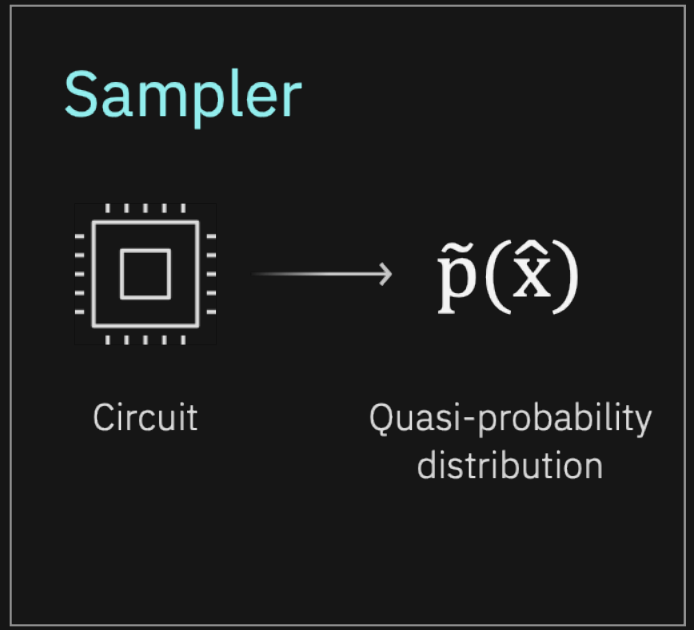


目次

Sampler primitive


```python
from qiskit.primitives import Sampler
from qiskit import QuantumCircuit

# Bell circuit
bell = QuantumCircuit(2)
bell.h(0)
bell.cx(0, 1)
bell.measure_all()

# initialization of the sampler
sampler = Sampler()

# Sampler runs a job on the Bell circuit
job = sampler.run(circuits=bell)
result = job.result()
print([q.binary_probabilities() for q in result.quasi_dists])
```

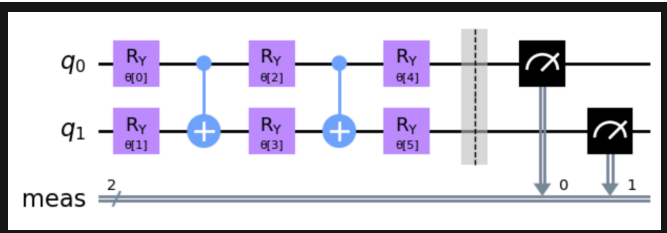
```
[{'00': 0.4999999999999999, '11': 0.4999999999999999}]
```

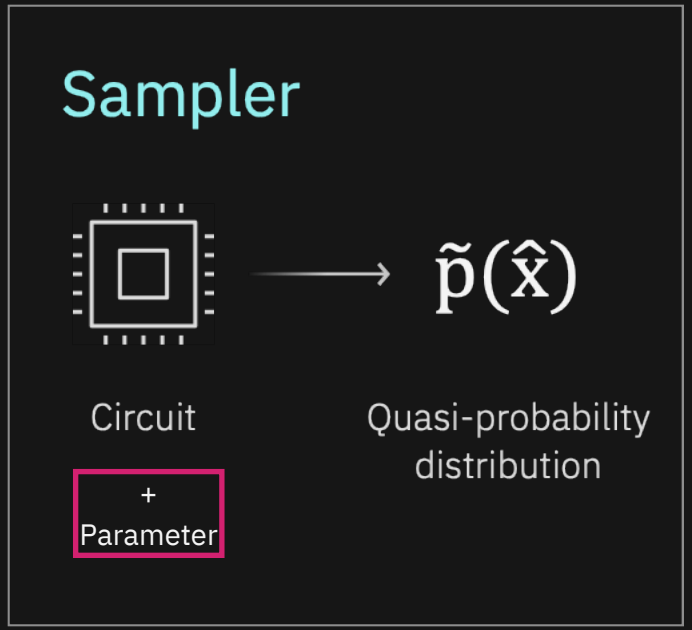


Sampler primitive (parameterized quantum circuit)


```python
from qiskit.circuit.library import RealAmplitudes

# two parameterized circuits
pqc = RealAmplitudes(num_qubits=2, reps=2)
pqc.measure_all()
pqc2 = RealAmplitudes(num_qubits=2, reps=3)
pqc2.measure_all()

theta1 = [0, 1, 1, 2, 3, 5]
theta2 = [0, 1, 2, 3, 4, 5, 6, 7]

# initialization of the sampler
sampler = Sampler()

# Sampler runs a job on the parameterized circuits
job2 = sampler.run(
    circuits=[pqc, pqc2],
    parameter_values=[theta1, theta2]
)
result = job2.result()
print([q.binary_probabilities() for q in result.quasi_dists])
```

```
[{'00': 0.1309248462975777, '01': 0.3608720796028449, '10': 0.0932486523205006, '11': 0.41
49544217790772}, {'00': 0.1880263994380416, '01': 0.6881971261189547, '10': 0.093262327205
8245, '11': 0.0305141472371799}]
```





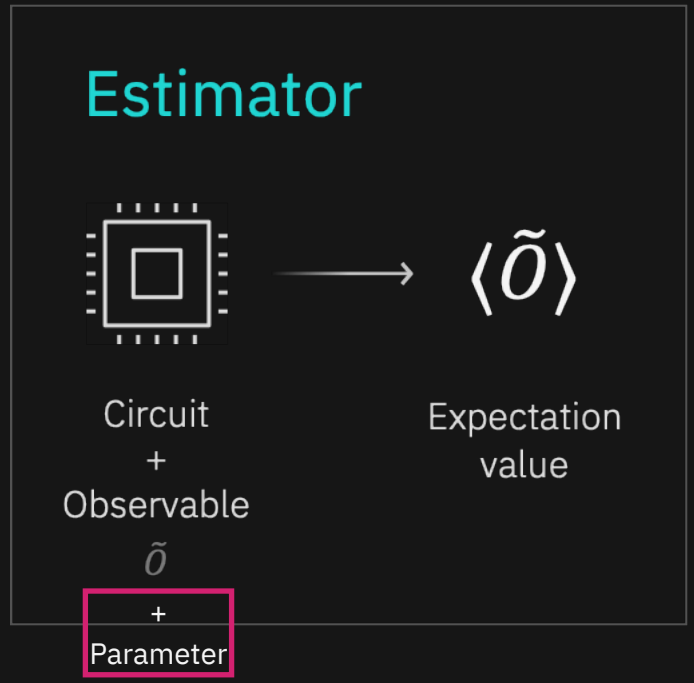
目次

Estimator primitive

IBM **Quantum**

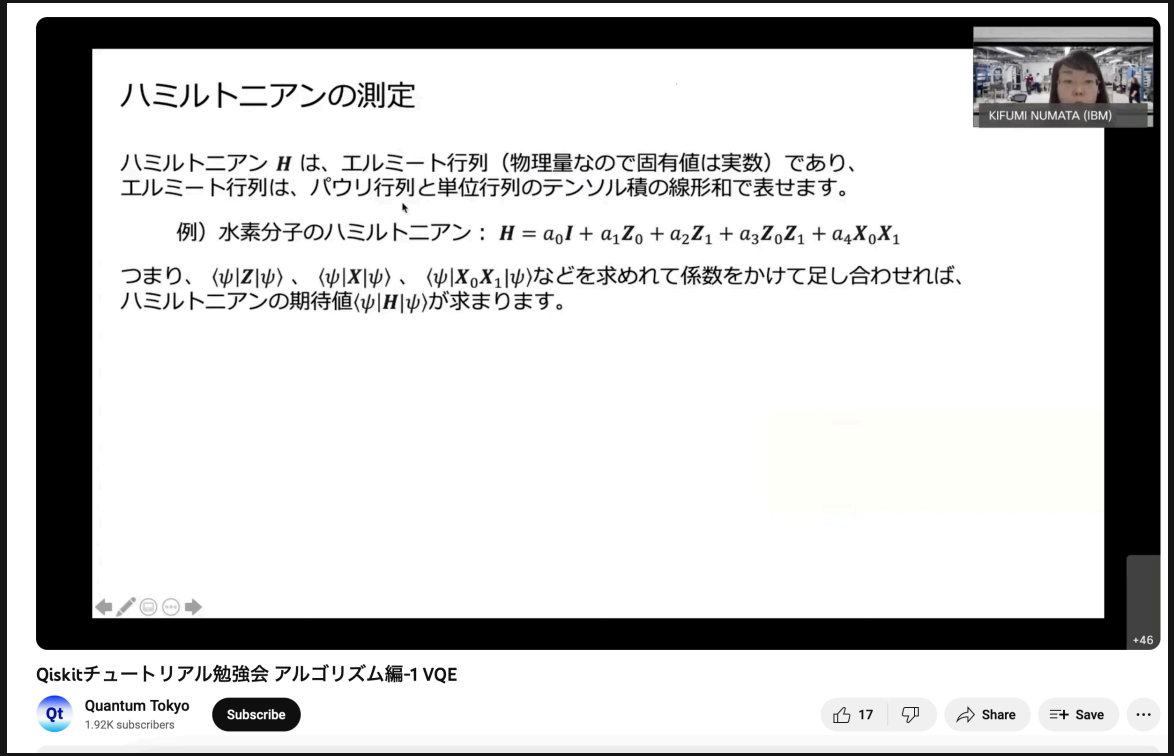
```python
from qiskit.primitives import Estimator
from qiskit.circuit.library import RealAmplitudes
from qiskit.quantum_info import SparsePauliOp

psi1 = RealAmplitudes(num_qubits=2, reps=2)
psi2 = RealAmplitudes(num_qubits=2, reps=3)

H1 = SparsePauliOp.from_list([("II", 1), ("IZ", 2), ("XI", 3)])
H2 = SparsePauliOp.from_list([("IZ", 1)])
H3 = SparsePauliOp.from_list([("ZI", 1), ("ZZ", 1)])

theta1 = [0, 1, 1, 2, 3, 5]
theta2 = [0, 1, 1, 2, 3, 5, 8, 13]
theta3 = [1, 2, 3, 4, 5, 6]

estimator = Estimator()

# calculate [ <psi1(theta1)|H1|psi1(theta1)> ]
job = estimator.run([psi1], [H1], [theta1])
result = job.result() # It will block until the job finishes.
print(f"Expectation values: {result.values}")

# calculate [ <psi1(theta1)|H1|psi1(theta1)>,
#             <psi2(theta2)|H2|psi2(theta2)>,
#             <psi1(theta3)|H3|psi1(theta3)> ]
job2 = estimator.run([psi1, psi2, psi1], [H1, H2, H3], [theta1, theta2, theta3])
result = job2.result()
print(f"Expectation values: {result.values}")
```

```
Expectation values: [1.55555728]
Expectation values: [ 1.55555728  0.17849238 -1.08766318]
```



期待値の測定について

IBM Quantum

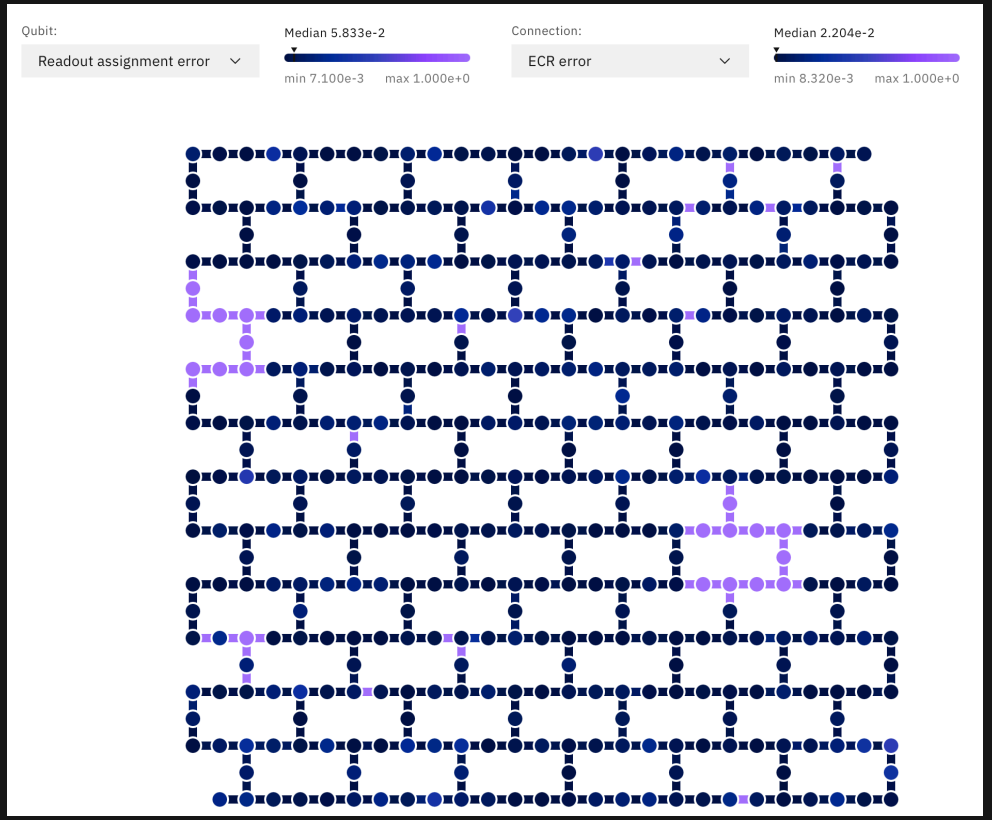


https://youtu.be/oSw3YLdW7Ls?si=FiLgO4i8FXLgp9Di&t=762

目次

Error in quantum processors

IBM **Quantum**



Type of Error

Random error

- ✓ Accuracy
- ✗ Precision

Take a large sampling number

Gate error
Readout error

Systematic error

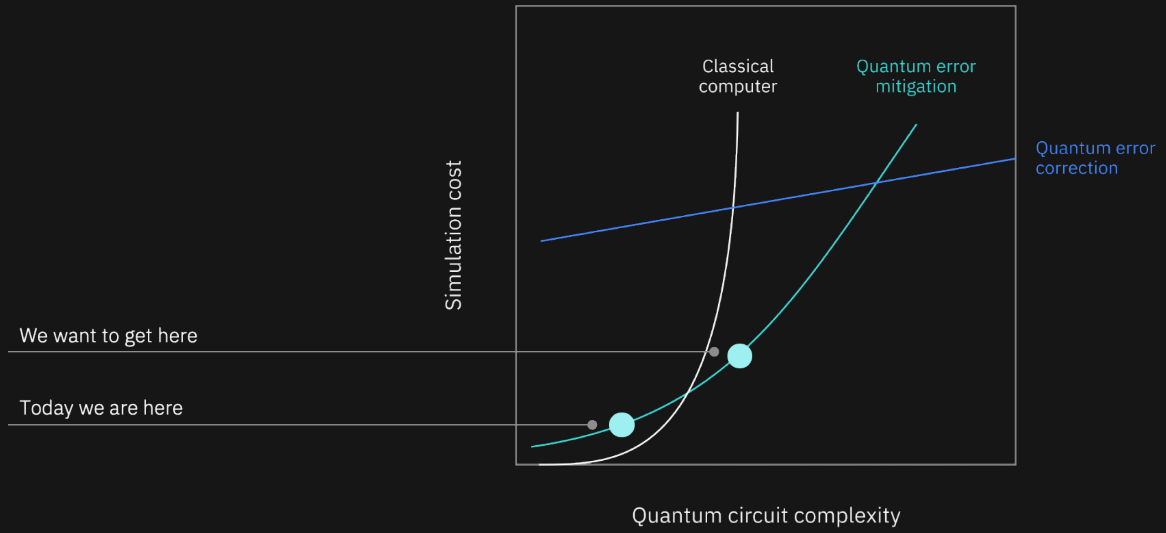
- ✗ Accuracy
- ✓ Precision

Error mitigation

Error mitigation and error suppression

Need to reduce the error.

- Quantum Error Correction (QEC)
- Quantum error mitigation and suppression



Error Suppression vs Error Mitigation

Error Suppression

- Technique that can tailor or reduce noise with no or minimal additional runtime cost
- Valid for any circuit and single shot case
- Does not require precise knowledge of noise processes

Error Mitigation

- Technique that removes noise from expectation value estimations
- Requires post-processing with additional cost
- Usually not valid in the single-shot case
- Requires various degrees of knowledge about noise process

Errors Suppression and Mitigation in the Qiskit IBM Runtime

Error Suppression Techniques

- Dynamical decoupling
- Measurement Pauli twirling
- 2-qubit gate Pauli twirling*

Error Mitigation Techniques

- Measurement error mitigation via M3 (Sampler) or TREX (Estimator)
- Gate error mitigation via Zero noise extrapolation (ZNE)
- Gate error mitigation via Probabilistic error mitigation (PEC)

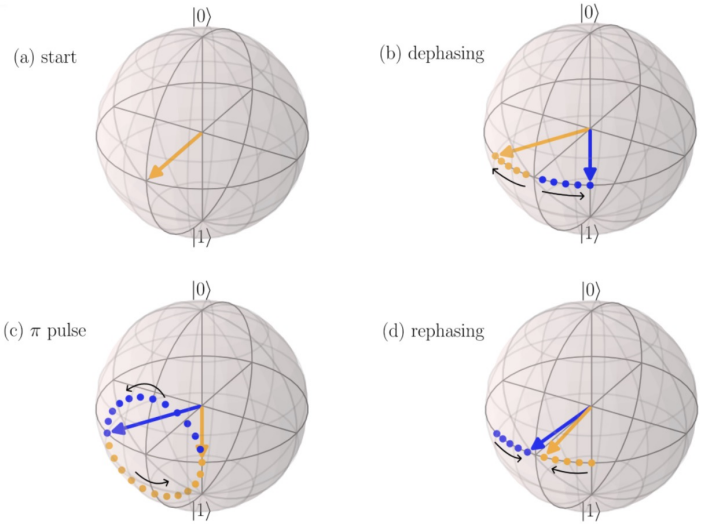
** Experimental branch*

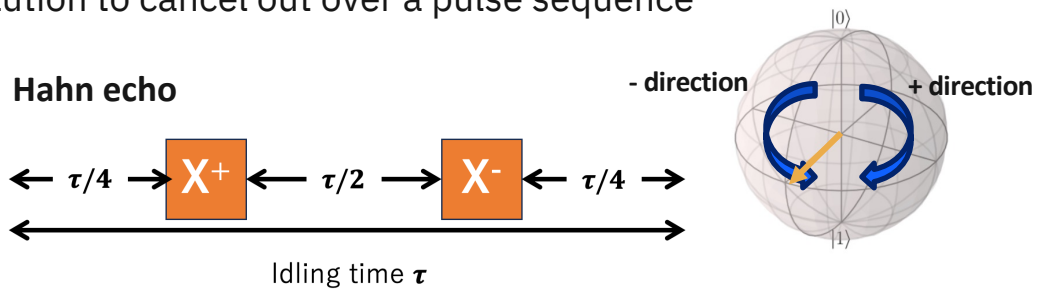
Error Suppression

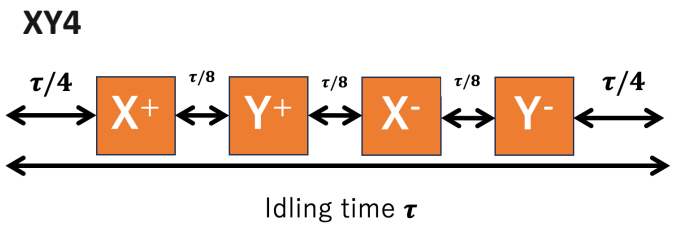
Dynamical Decoupling (DD)

Viola, Knill, Lloyd, PRL 82 (1999)

- DD is a family of control techniques designed to suppress decoherence resulting from unwanted *coherent* evolution
- Achieves this by modifying the coherent evolution to cancel out over a pulse sequence



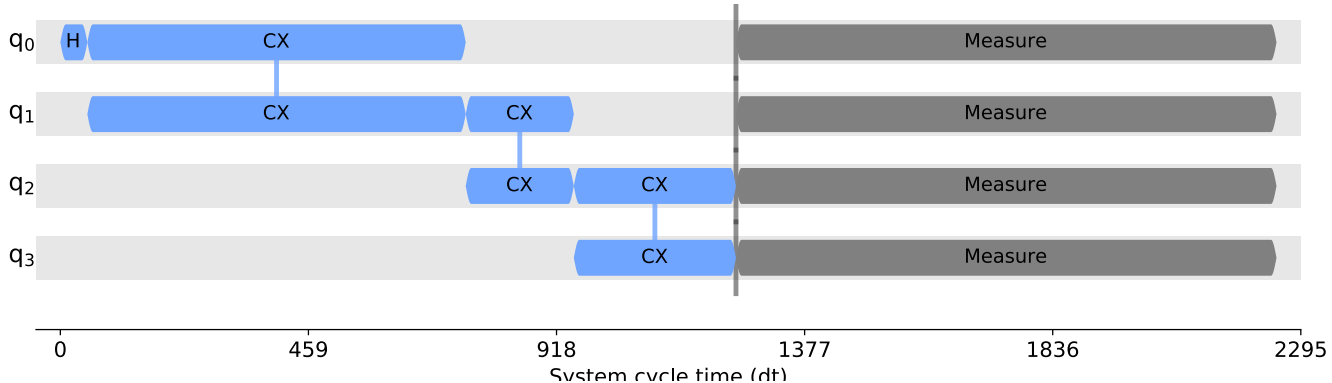




Dynamical Decoupling (DD)

DD implement in Qiskit via transpilation passes which identify large idle spaces in circuits and inserts DD sequences into them

- Achieves this by modifying the coherent evolution to cancel out over a pulse sequence

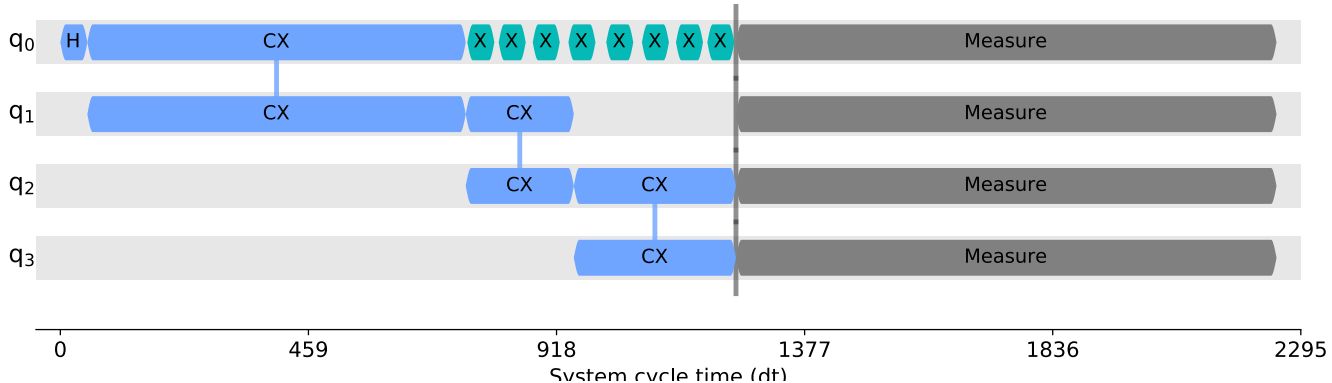


https://qiskit.org/documentation/stubs/qiskit.transpiler.passes.DynamicalDecoupling.html

Dynamical Decoupling (DD)

DD implement in Qiskit via transpilation passes which identify large idle spaces in circuits and inserts DD sequences into them

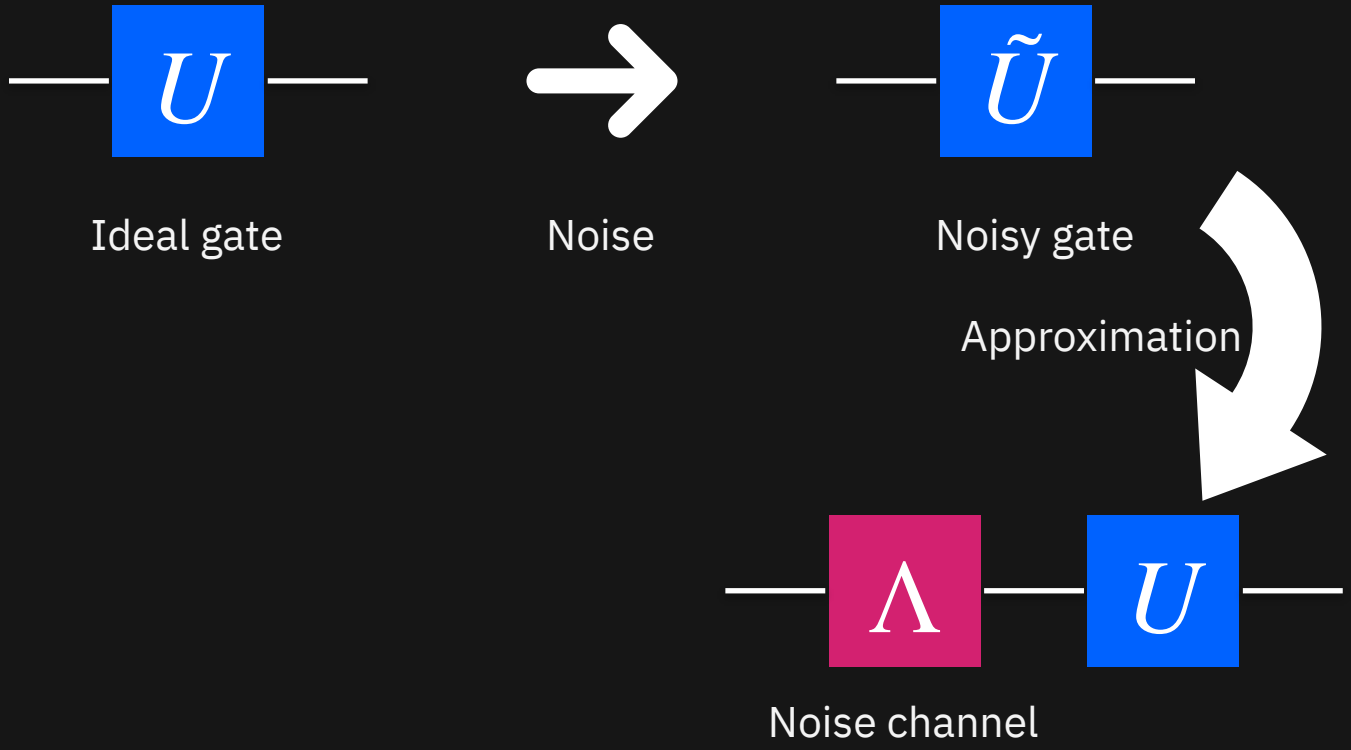
- Achieves this by modifying the coherent evolution to cancel out over a pulse sequence



https://qiskit.org/documentation/stubs/qiskit.transpiler.passes.DynamicalDecoupling.html

Pauli Twirling

Gate error (TPCP map)



Quantum Channels

General description of noise in quantum operations

$$\Lambda(\rho) = \sum_i K_i \rho K_i^+$$

Noise gates are described as an ideal gate followed by an error quantum channel $U_{noise} = U_{ideal} \circ \Lambda$

One representation of quantum channels is the Pauli-transfer matrix (PTM) R

$$\Lambda(\rho) = \sum_{ij} R_{ij}\ P_i Tr[P_j \rho]$$

A special class of error channels are Pauli Channels, which are quantum channels that have a diagonal PTM

$$\Lambda_{PT}(\rho) = \sum_{ii} R_{ii}\ P_i Tr[P_i \rho]$$

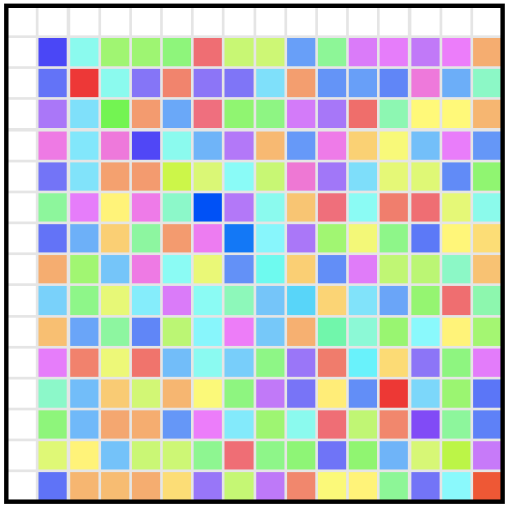
Pauli Twirling

MR Geller, Z Zhou PRA 88 012314 (2013)

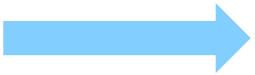
JJ Wallman, J Emerson, PRA 94, 052325 (2016)

- An error suppression technique for tailoring noise in individual quantum channels
- Converts a general quantum channel into a Pauli channel by averaging over randomly sampled circuits

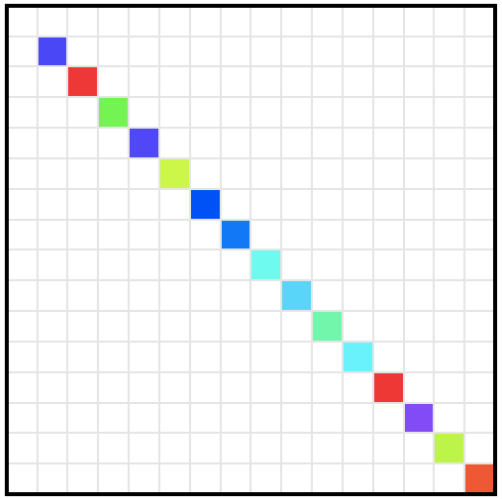
General Channel PTM



Pauli Twirl



Twirled Pauli Channel PTM



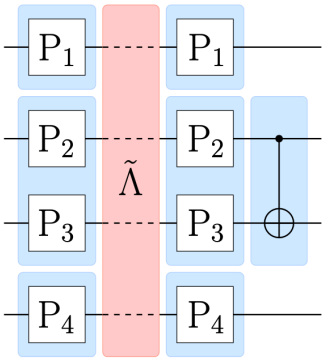
Pauli Twirling

MR Geller, Z Zhou PRA 88 012314 (2013)

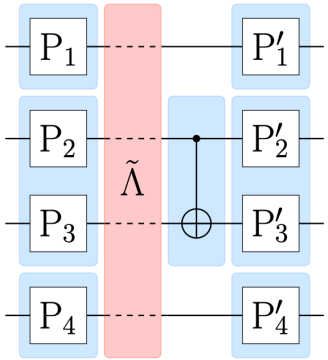
JJ Wallman, J Emerson, PRA 94, 052325 (2016)

- We apply Pauli twirling to 2-qubit Clifford gates in our circuits (eg CNOT, CZ, ECR) and end of circuit measurements
- Sampling 1-qubit Paulis and averaging over randomizations
- For dense circuits this does not increase the depth or gate count of the circuit

Pauli Twirled Noise

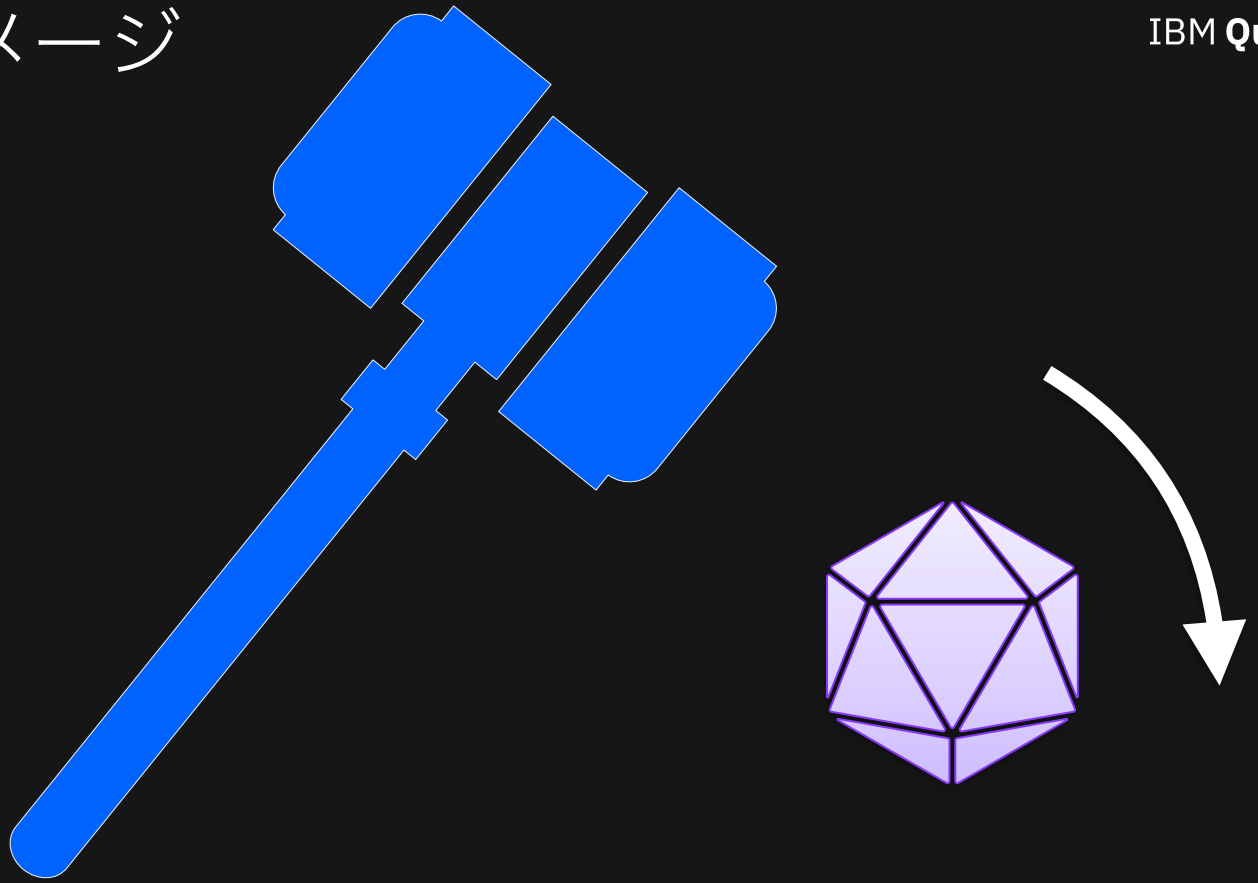


Sampled Paulis to implement Twirl



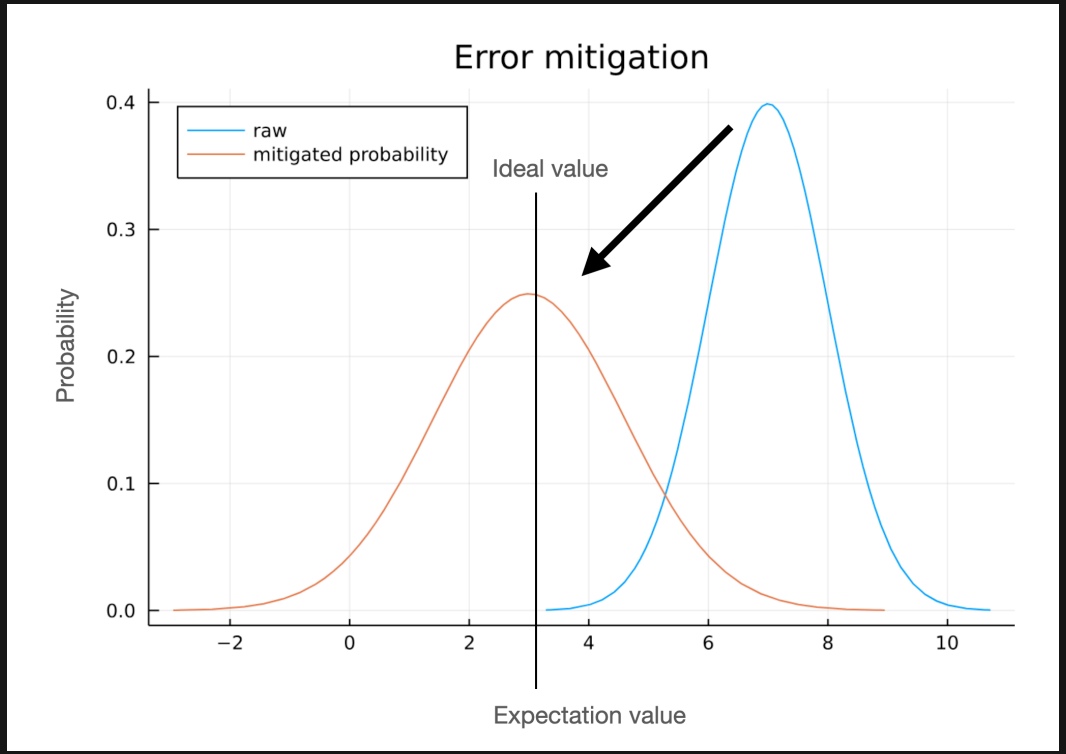
Pauli twirling doesn't change circuit unitary or number of 2-qubit gates

Twirlingのイメージ



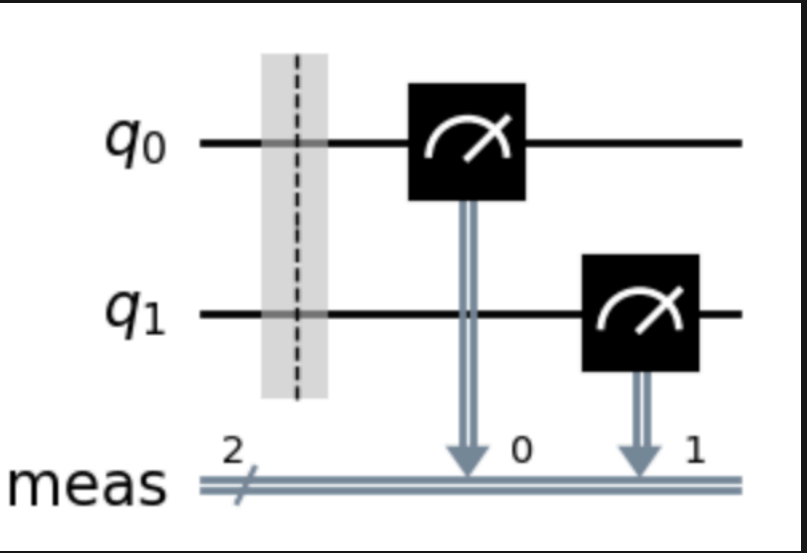
Error Mitigation

Error mitigation



Twirled Measure Error Mitigation

Readout error



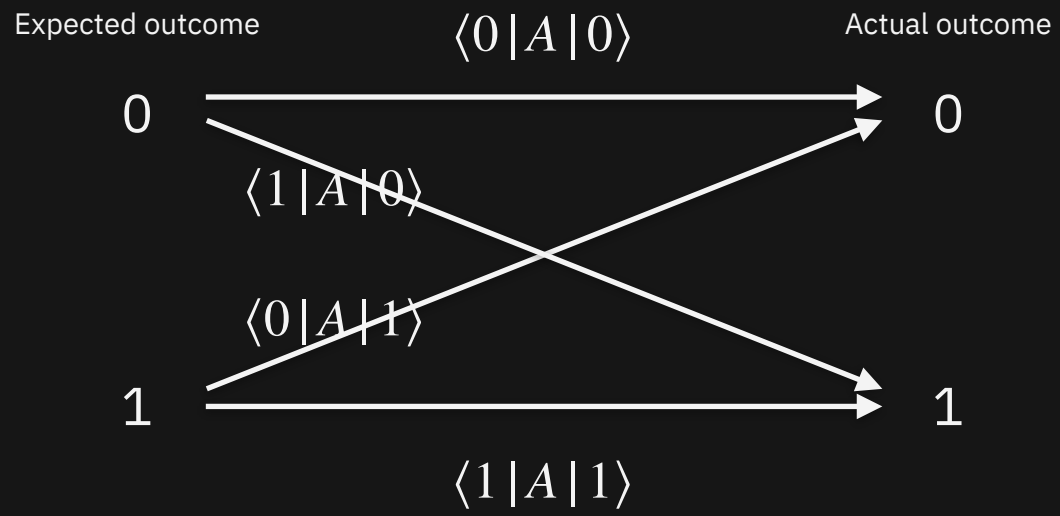
Readout

```
result = Sampler("ibmq_qasm_simulator").run(qc).result()
result
```

```
SamplerResult(quasi_dists=[{0: 1.0}], metadata=[{'shots': 4000}])
```

If readout error exists, the wrong result would be obtained. e.g. {“00”: 0.9, “01”: 0.05, “10”: 0.04, “11”: 0.01}

Readout error



A matrix (assignment matrix, noise map) characterizes readout errors

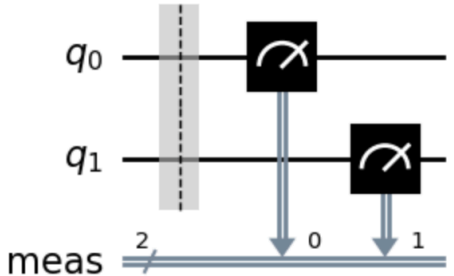
How to characterize readout error?

IBM **Quantum**

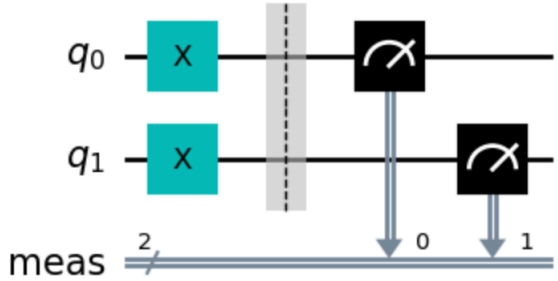
Calibration experiment

```
from qiskit_experiments.library import LocalReadoutError
exp = LocalReadoutError(qubits)
```

```
exp.circuits()[0].draw()
```




```
exp.circuits()[1].draw()
```



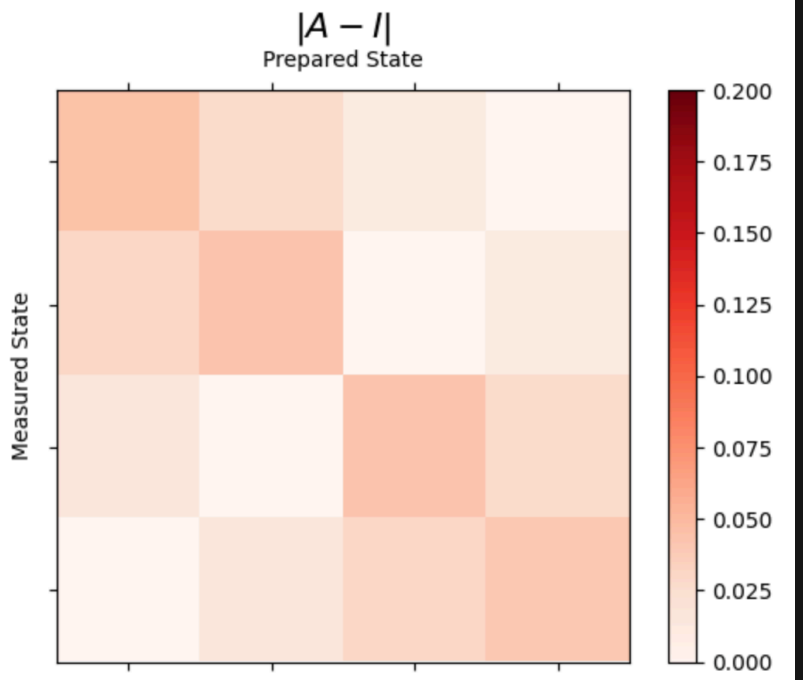
Ref. https://qiskit.org/ecosystem/experiments/manuals/measurement/readout_mitigation.html

Run calibration experiment

IBM **Quantum**

```
exp.analysis.set_options(plot=True)
result = exp.run(backend)
```

```
result.figure(0)
```




```
print(mitigator.assignment_matrix(0))
```

```
[[0.98242188 0.02734375]
 [0.01757812 0.97265625]]
```

```
print(mitigator.assignment_matrix(1))
```

```
[[0.98535156 0.01855469]
 [0.01464844 0.98144531]]
```

```
print(mitigator.assignment_matrix())
```

```
[[9.68030930e-01 2.69432068e-02 1.82285309e-02 5.07354736e-04]
 [1.73206329e-02 9.58408356e-01 3.26156616e-04 1.80473328e-02]
 [1.43909454e-02 4.00543213e-04 9.64193344e-01 2.68363953e-02]
 [2.57492065e-04 1.42478943e-02 1.72519684e-02 9.54608917e-01]]
```

Ref. https://qiskit.org/ecosystem/experiments/manuals/measurement/readout_mitigation.html

Mitigation matrix


```
print(mitigator.mitigation_matrix())
```

```
[[ 1.03383529e+00 -2.90636426e-02 -1.95451447e-02  5.49461899e-04]
 [-1.86837702e-02  1.04421516e+00  3.53225507e-04 -1.97413811e-02]
 [-1.54303774e-02  4.33785710e-04  1.03795005e+00 -2.91793187e-02]
 [ 2.78862242e-04 -1.55853009e-02 -1.87581335e-02  1.04837124e+00]]
```

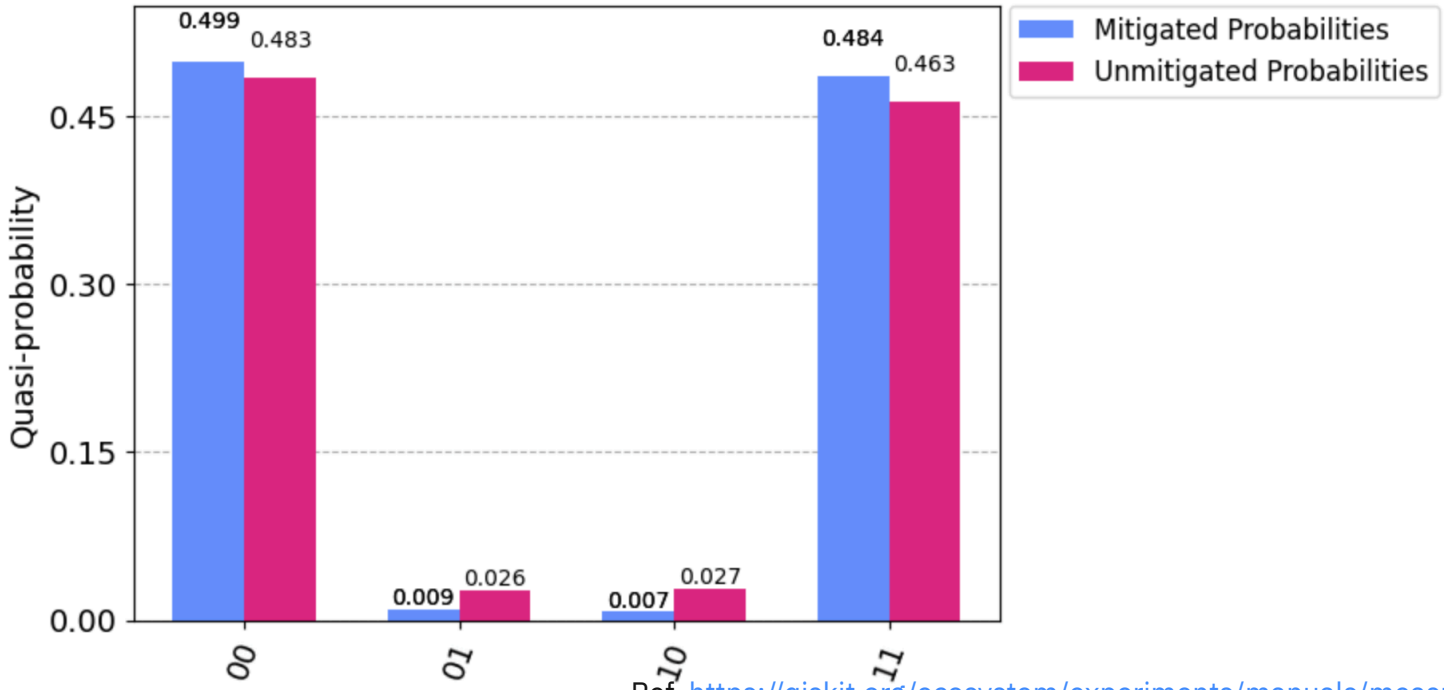
$$A^{-1}$$

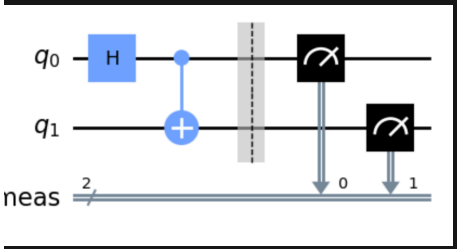
Ref. https://qiskit.org/ecosystem/experiments/manuals/measurement/readout_mitigation.html

Example of readout error mitigation

IBM **Quantum**

```
counts = backend.run(qc, shots=shots, seed_simulator=42, method="density_matrix").result().get_counts()
unmitigated_probs = {label: count / shots for label, count in counts.items()}
mitigated_quasi_probs = mitigator.quasi_probabilities(counts)
mitigated_probs = (mitigated_quasi_probs.nearest_probability_distribution().binary_probabilities())
plot_histogram([mitigated_probs, unmitigated_probs], legend=['Mitigated Probabilities', 'Unmitigated Probabilities'])
```



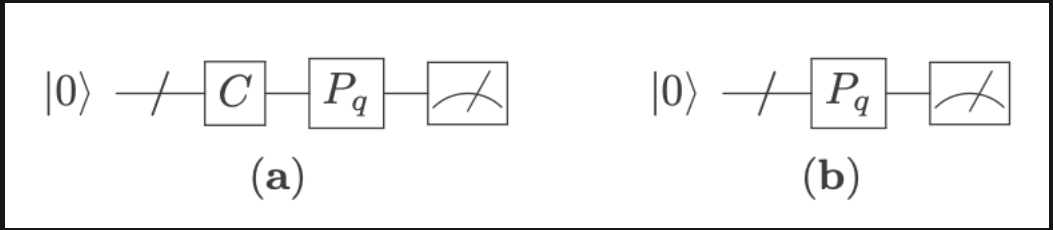


Ref https://qiskit.org/ecosystem/experiments/manuals/measurement/readout_mitigation.html

Twirled Readout Error eXtinction (TREX)

IBM **Quantum**

1. Symmetrize measurement error by twirling.
2. Divide $\widetilde{\langle Z_w \rangle_\rho}$ by $\widetilde{\langle Z_w \rangle_0}$. $\dfrac{\widetilde{\langle Z_\omega \rangle_\rho}}{\widetilde{\langle Z_\omega \rangle_0}}$

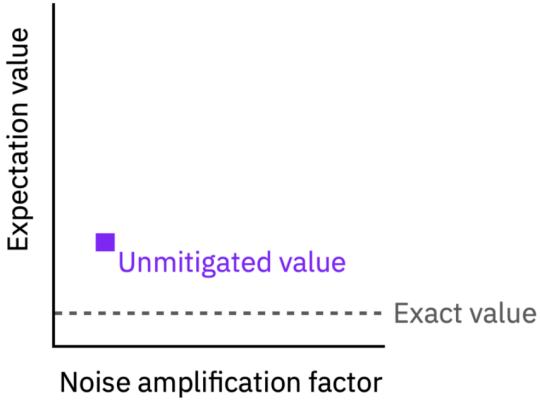


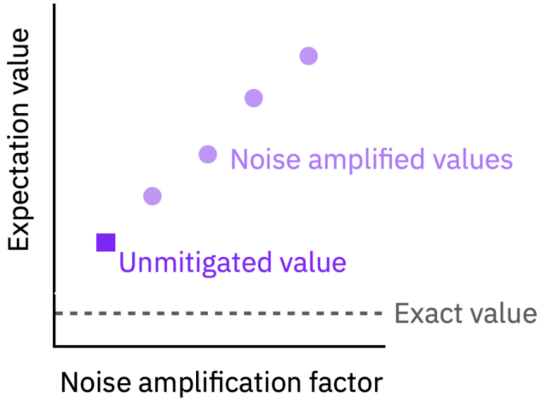
Ewout van den Berg, Zlatko K. Minev, and Kristan Temme Phys. Rev. A **105**, 032620

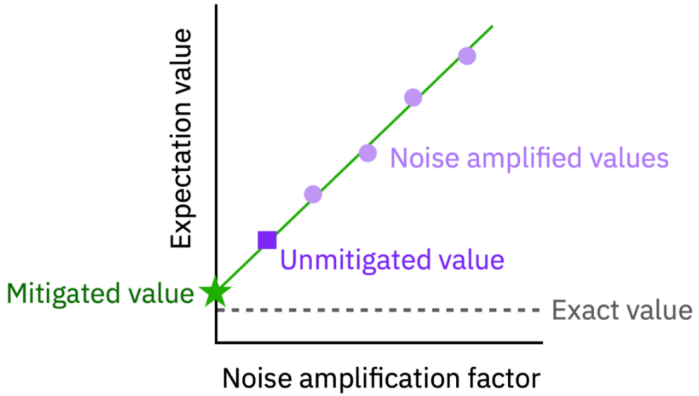
Zero Noise Extrapolation (ZNE)

Zero noise extrapolation

1. Amplify circuit noise for several noise factors
2. Run every noise amplified circuit
3. Extrapolate back to the zero noise limit



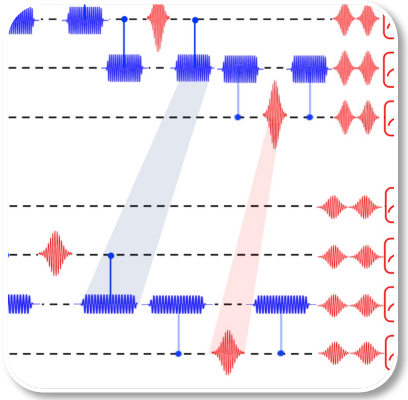




How to scale noise?

Pulse Stretching

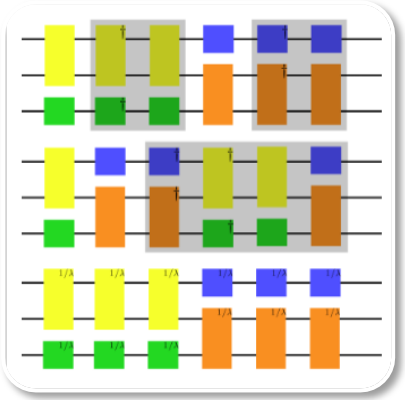
- Scale pulse duration via calibration.



A Kandala et al. Nature (2019)

Gate Folding

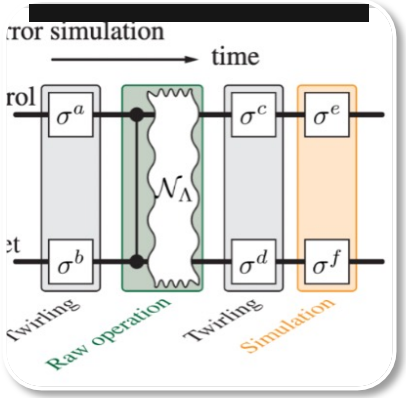
- Repeat gates in identity cycles $U \rightarrow U(U^{-1}U)^{(\lambda-1)/2}$



K Shultz et al. PRA (2022)

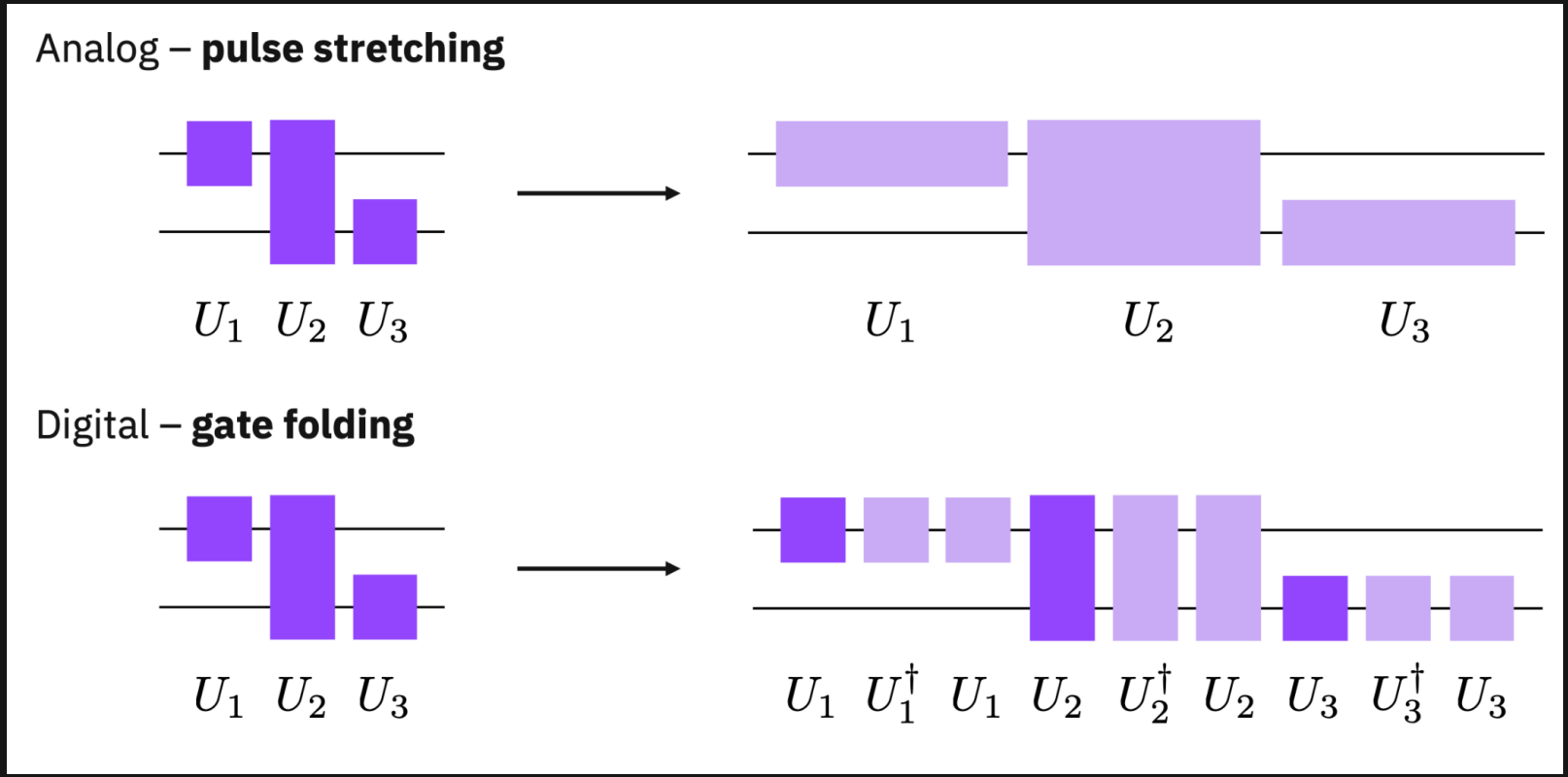
Random Sampling

- Add noise via sampling Pauli channels



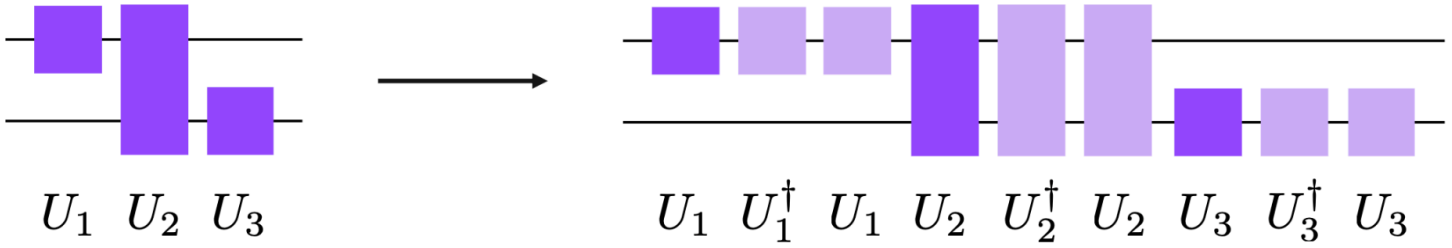
Y Li, SC Benjamin, PRX (2017)

Noise amplification

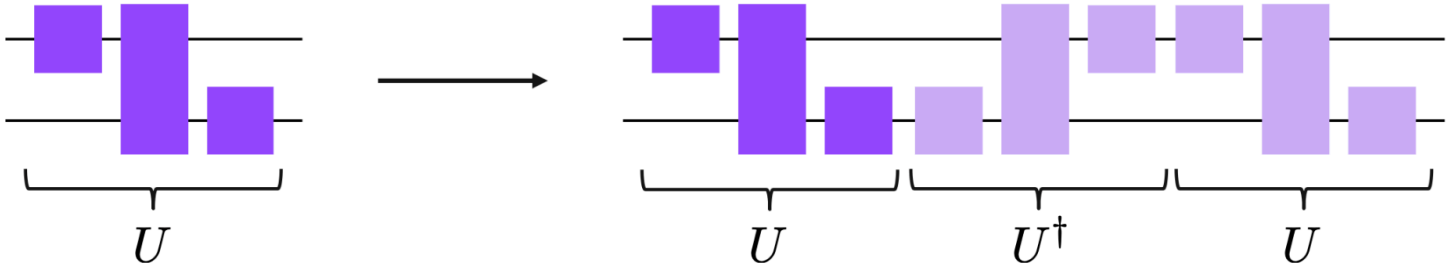


Noise amplification

Local folding – repeatedly folding the **gates** inside the circuit



Global folding – repeatedly folding the whole **circuit**



Probabilistic error amplification (PEA)

Article

Evidence for the utility of quantum computing before fault tolerance

https://doi.org/10.1038/s41586-023-06096-3

Received: 24 February 2023

Accepted: 18 April 2023

Published online: 14 June 2023

Open access

Check for updates

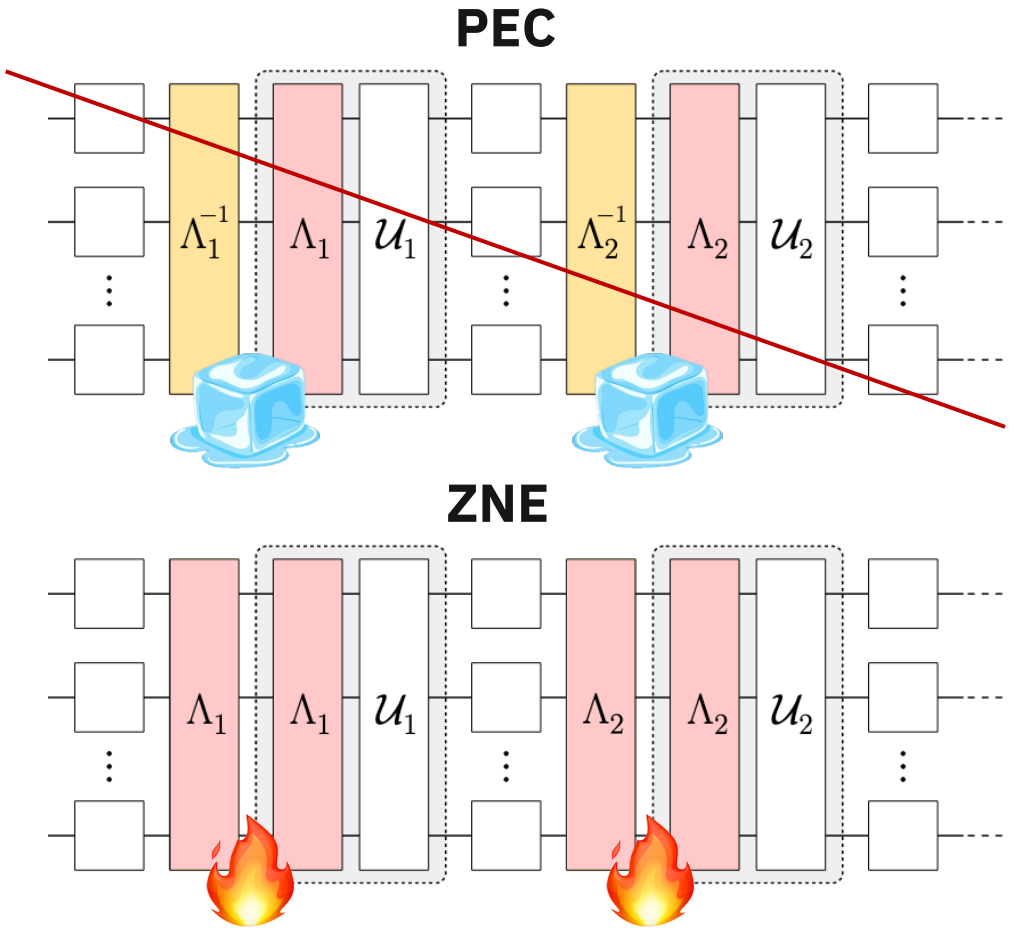
Youngseok Kim[1,6]✉, Andrew Eddins[2,6]✉, Sajant Anand[3], Ken Xuan Wei[1], Ewout van den Berg[1], Sami Rosenblatt[1], Hasan Nayfeh[1], Yantao Wu[3,4], Michael Zaletel[3,5], Kristan Temme[1] & Abhinav Kandala[1]✉

Quantum computing promises to offer substantial speed-ups over its classical counterpart for certain problems. However, the greatest impediment to realizing its full potential is noise that is inherent to these systems. The widely accepted solution to this challenge is the implementation of fault-tolerant quantum circuits, which is out of reach for current processors. Here we report experiments on a noisy 127-qubit processor and demonstrate the measurement of accurate expectation values for circuit volumes at a scale beyond brute-force classical computation. We argue that this represents evidence for the utility of quantum computing in a pre-fault-tolerant era. These experimental results are enabled by advances in the coherence and calibration of a superconducting processor at this scale and the ability to characterize[1] and controllably manipulate noise across such a large device. We establish the accuracy of the measured expectation values by comparing them with the output of exactly verifiable circuits. In the regime of strong entanglement, the quantum computer provides correct results for which leading classical approximations such as pure-state-based 1D (matrix product states, MPS) and 2D (isometric tensor network states, isoTNS) tensor network methods[2,3] break down. These experiments demonstrate a foundational tool for the realization of near-term quantum applications[4,5].

ZNE w/ probabilistic amplification

- Use PEC sparse Lindblad learning to identify noise models
- Scale noise by sampling more noise (probabilistic error amplification) instead of mitigation

Y Kim et al. Evidence for the utility of quantum computing before fault tolerance, Nature 618 (2023)

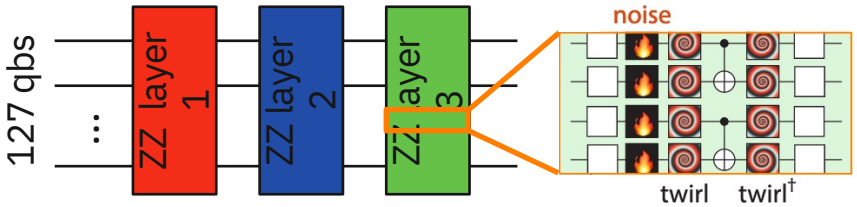


ZNE w/ probabilistic amplification (127-qubits)

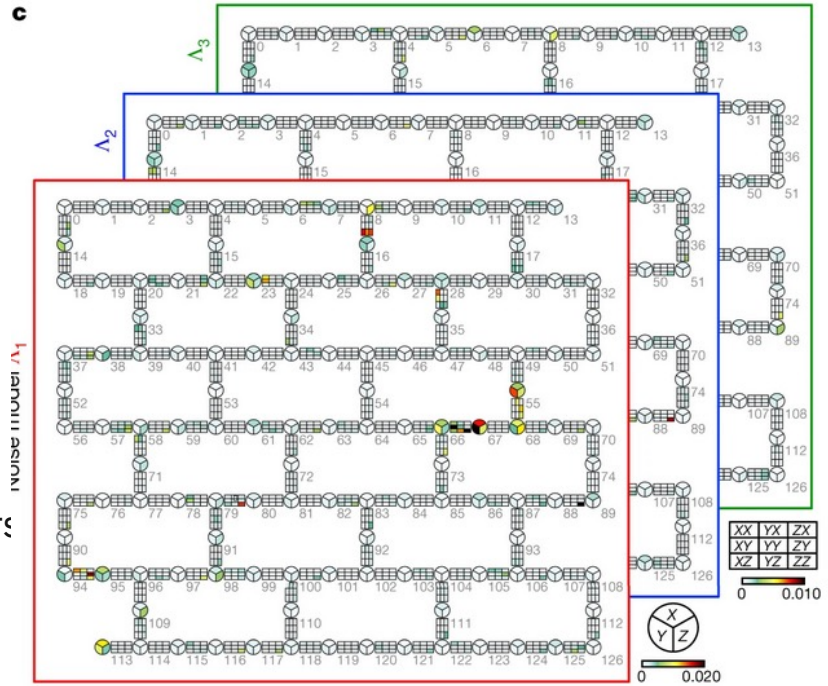
IBM **Quantum**

Y Kim et al. Evidence for the utility of quantum computing before fault tolerance, Nature, (2023)

- 127-qubit simulation of trotter quench dynamics of 2D-ising lattice
- $R_{ZZ}(\frac{\pi}{2})$ on all neighbor pairs = 3 depth-1 layers



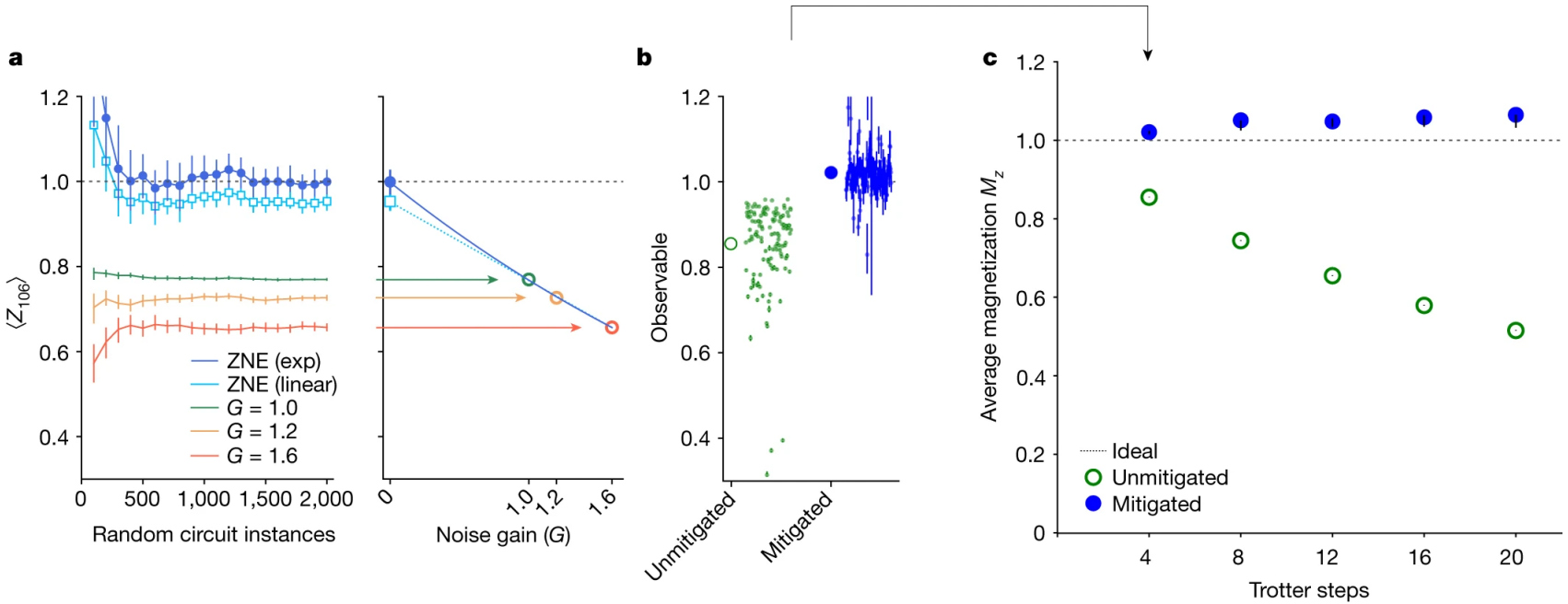
- IBM Kyiv**,** Eagle processor (127-qubit)
- Several hours to learn all Pauli noise models for 3-layers



ZNE w/ probabilistic amplification (127-qubits)

IBM **Quantum**

Y Kim et al. Nature 618 (2023)

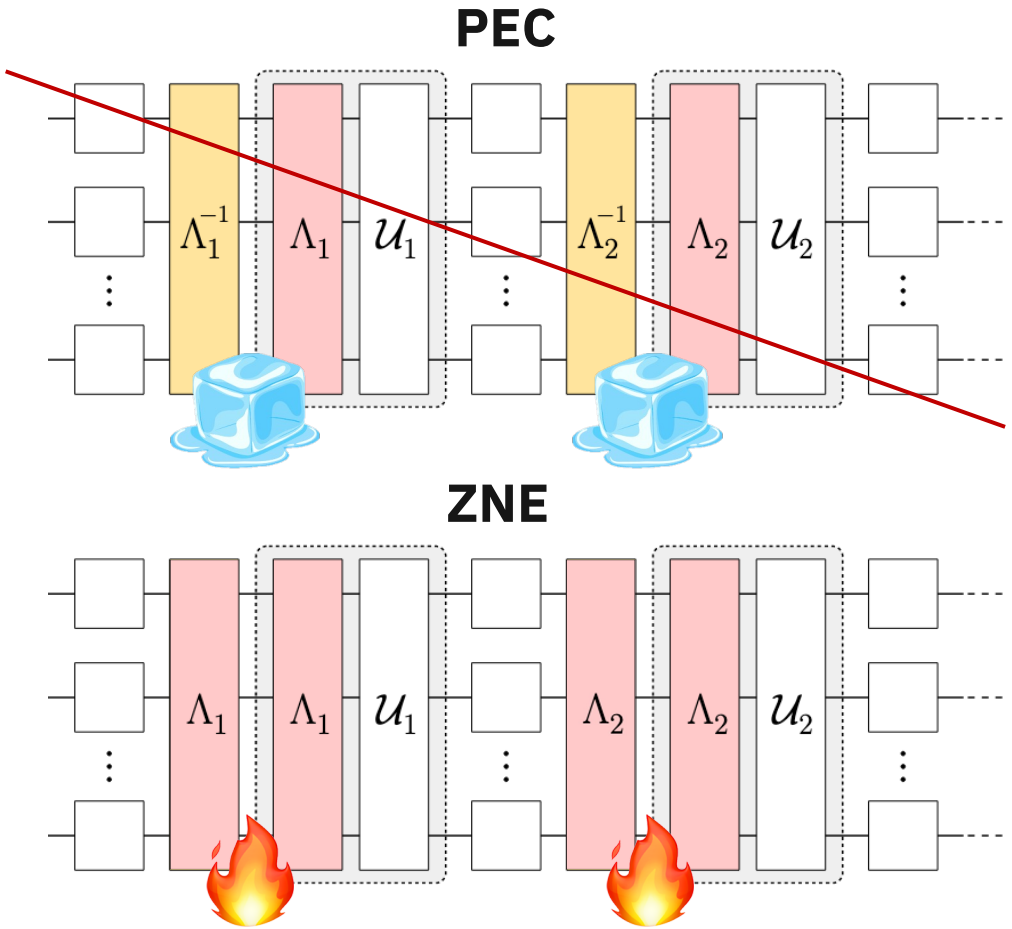


Probabilistic Error Cancellation (PEC)

ZNE w/ probabilistic amplification

- Use PEC sparse Lindblad learning to identify noise models
- Scale noise by sampling more noise (probabilistic error amplification) instead of mitigation

Y Kim et al. Evidence for the utility of quantum computing before fault tolerance, Nature 618 (2023)



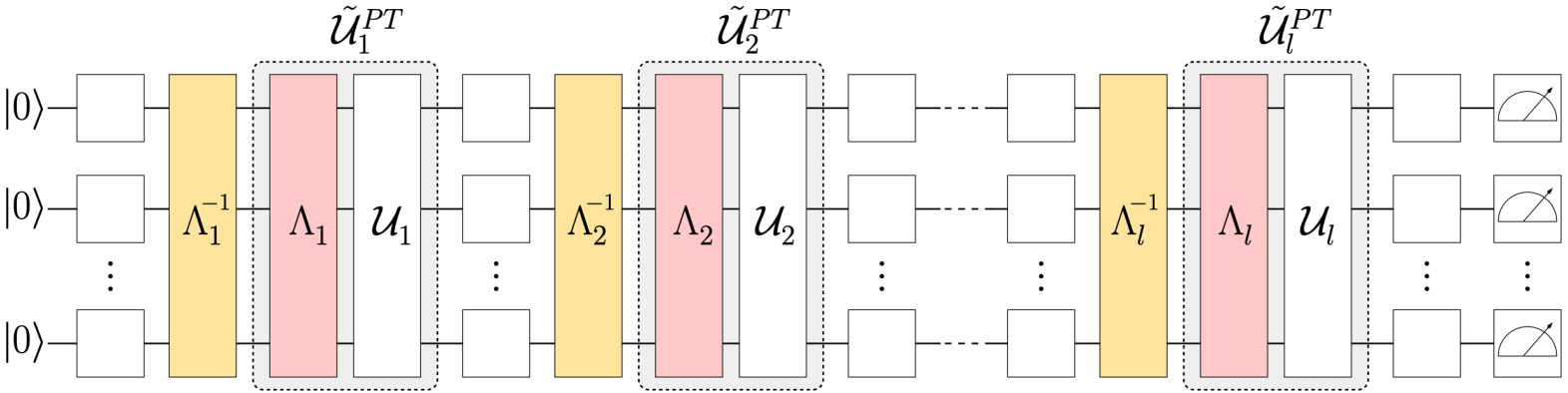
Probabilistic Error Cancellation (PEC)

IBM **Quantum**

K Temme, S Bravyi, JM Gambetta, PRL 119 180509 (2017)

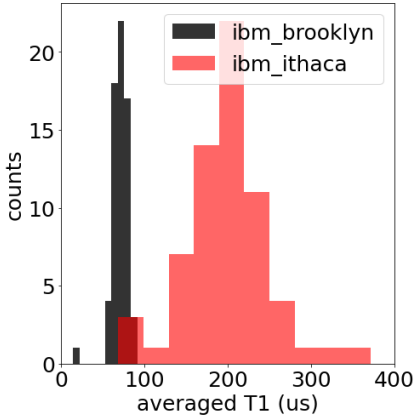
E van den Berg et al. arXiv:2201.09866 (2022)

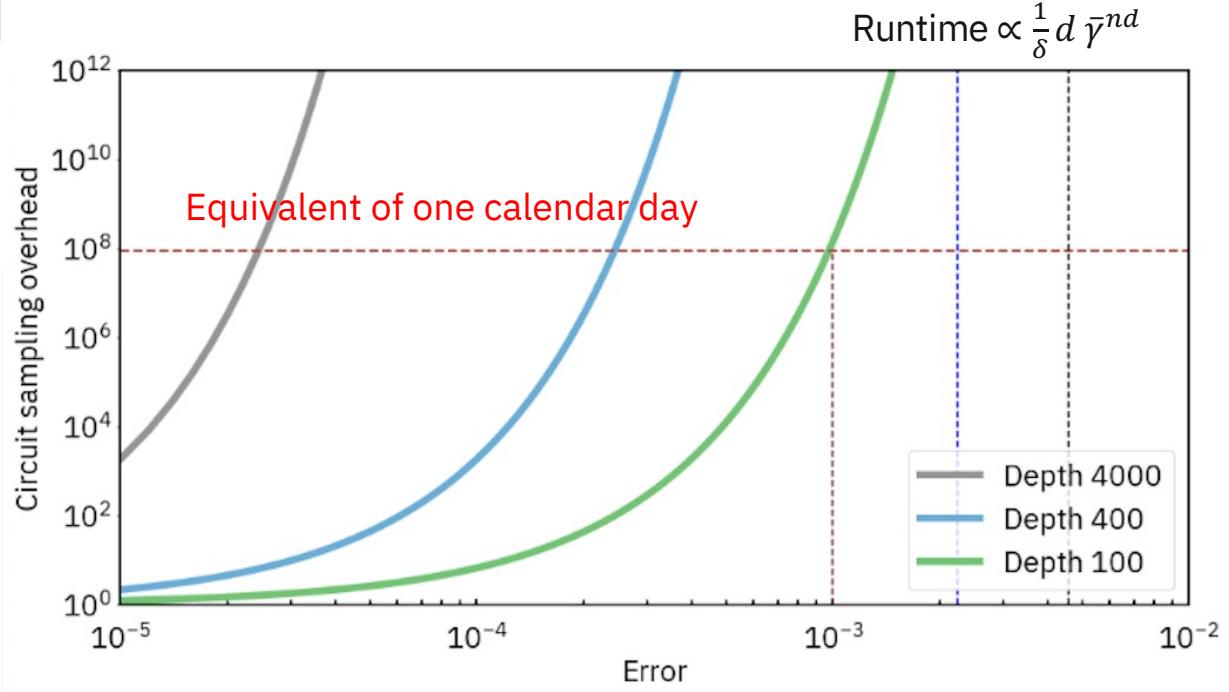
- If $\widetilde{\Lambda}_i$ is invertible we can would like to insert $\widetilde{\Lambda}_i^{-1}$ to undo noise
- $\widetilde{\Lambda}_i^{-1}$ is not a physical operation
- If it can be decomposed into a probabilistic sum of unitaries we could approximate it via sampling and averaging



PEC mitigation overhead

Device	$\bar{\gamma}$ (measured on 10Q)
Brooklyn	1.038
Ithaca	1.024
Prague	1.012





Resilience level

Resilience Level	Estimator	Sampler
0	None	None
1 [Default]	Twirled Readout Error eXtinction (TREX)	M3
2	Zero Noise Extrapolation (ZNE)	—
3	Probabilistic Error Cancellation (PEC)	—

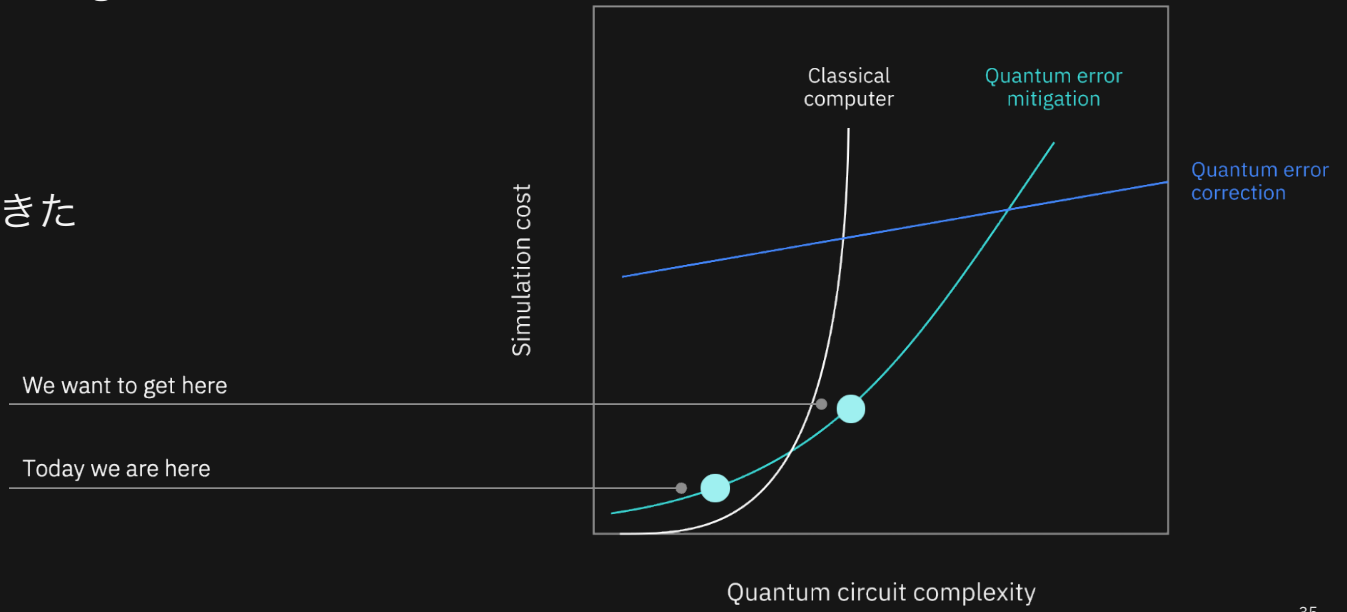
https://qiskit.org/ecosystem/ibm-runtime/how_to/error-mitigation.html

Conclusion remarks

IBM **Quantum**

- Error mitigation is essential tool for noisy quantum computer.
- Along with improved methods of mitigation, there is a need to explore applications for quantum advantage.

100量子ビット時代はやってきた



おすすめチュートリアル

Qiskit

基本の書き方

・Sampler：https://qiskit.org/ecosystem/ibm-runtime/locale/ja_JP/tutorials/how-to-getting-started-with-sampler.html

・Estimator：https://qiskit.org/ecosystem/ibm-runtime/locale/ja_JP/tutorials/how-to-getting-started-with-estimator.html

実機で実行

・CHSH：https://qiskit.org/ecosystem/ibm-runtime/locale/ja_JP/tutorials/chsh_with_estimator.html

・Grover：https://qiskit.org/ecosystem/ibm-runtime/locale/ja_JP/tutorials/grover_with_sampler.html

エラー抑制・緩和

https://qiskit.org/ecosystem/ibm-runtime/locale/ja_JP/tutorials/Error-Suppression-and-Error-Mitigation.html

アプリケーションの書き方

・VQE：https://qiskit.org/ecosystem/ibm-runtime/locale/ja_JP/tutorials/vqe_with_estimator.html

・OAOA：https://qiskit.org/ecosystem/ibm-runtime/locale/ja_JP/tutorials/qaoa_with_primitives.html

IBM **Quantum**

