Execution on noisy quantum hardware

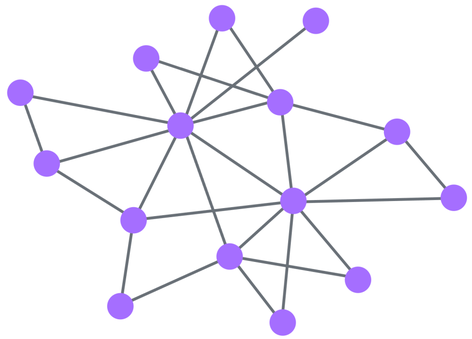
Kevin J. Sung
Quantum Algorithm Engineer
IBM

IBM **Quantum**

Qiskit patterns

Step 1

Map classical inputs to a quantum problem



Step 2

Optimize problem for quantum execution.

```
PassManager([UnitarySynthesis(),
             BasisTranslator(),
             EnlargeWithAncilla(),
             AISwap(),
             Collect1qRuns(),
             Optimize1qGates(),
             Collect2qBlocks(),
             ConsolidateBlocks()])
```

Step 3

Execute using Qiskit Runtime Primitives.

Sampler

circuit($\vec{\theta}$)

000101...,
110110...

bit-strings

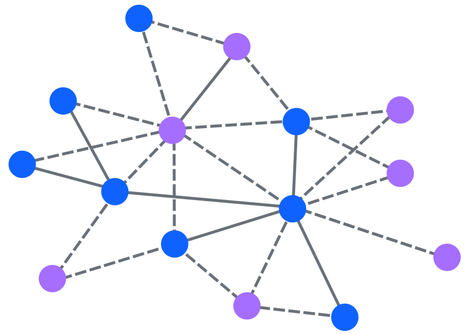
Estimator

circuit($\vec{\theta}$) +
observable $\hat{O}$

expectation value

Step 4

Post-process, return result in classical format.



Execute using Qiskit Runtime Primitives

Step 3

Execute using Qiskit Runtime Primitives.

Sampler

circuit($\vec{\theta}$)

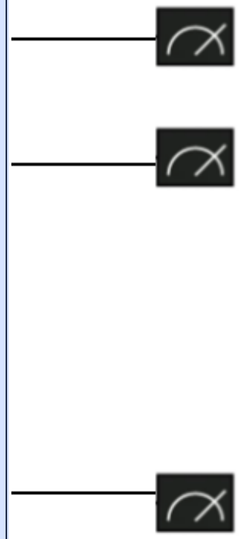
`000101...,`
`110110...`

bit-strings

Estimator

circuit($\vec{\theta}$) + observable $\hat{O}$

expectation value



Accessing the output of the quantum circuit

Quantum computers are noisy!

We need ways to recover a better signal:

1. Limit the amount of noise
2. Clean the signal by filtering the noise out

This is accomplished by:

- Run modified noisy quantum computations
- Process collected outputs on a classical computer

The primitives include options to implement error suppression and error mitigation.

Outline

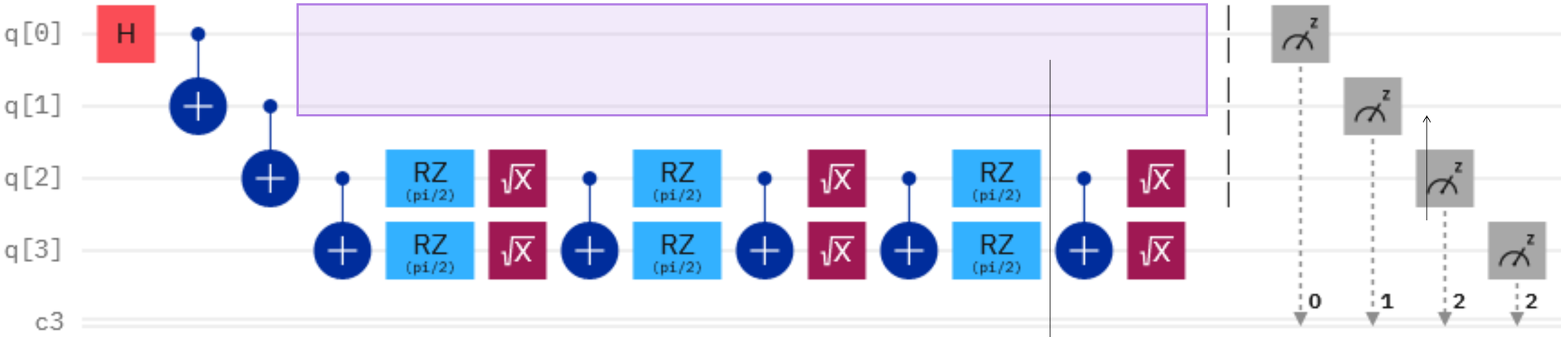
1 Noise in quantum systems

2 Error suppression techniques

3 Error mitigation techniques

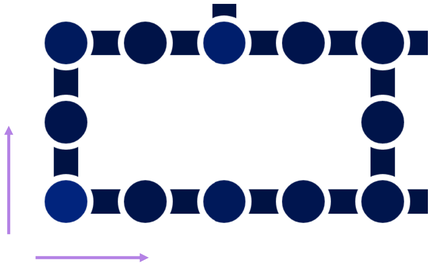
4 Combining techniques

Noise in quantum systems



Bath/system coupling

Environmental noise: relevant for deep sparse circuits



ibm_torino

OpenQASM 3

Details

133 Qubits	Status:	Online	Median CZ error: 4.384e-3
0.8% EPLG	Total pending jobs:	43 jobs	Median SX error: 3.161e-4
3.8K CLOPS	Processor type ⓘ:	Heron r1	Median readout error: 1.770e-2
	Version:	1.0.16	Median T1: 175.89 us
	Basis gates:	CZ, ID, RZ, SX, X	Median T2: 134.83 us
	Your instance usage:	0 jobs	

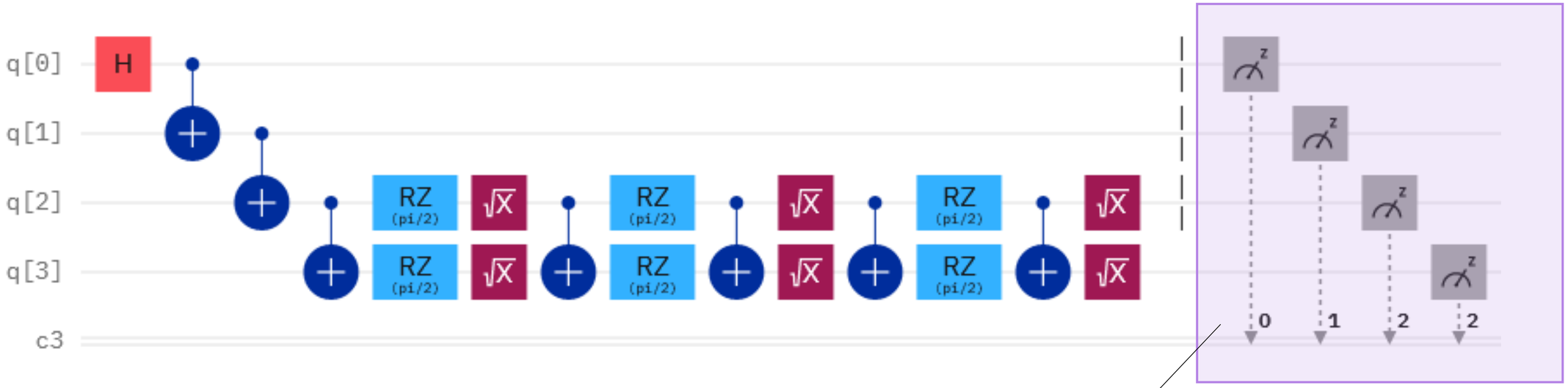
Decoherence

Information loss over time.

Crosstalk

Idle qubits interact with their neighbors. (Difficult to predict but less severe in heron devices!)

Noise in quantum systems



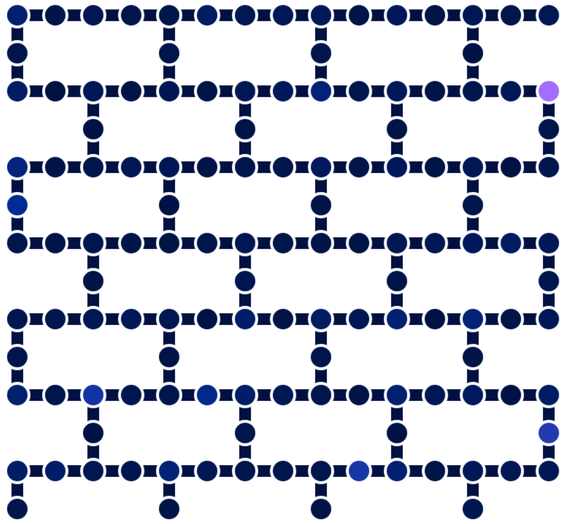
ibm_torino

OpenQASM 3

Details

133 Qubits	Status:	Online	Median CZ error: 4.384e-3
0.8% EPLG	Total pending jobs:	43 jobs	Median SX error: 3.161e-4
3.8K CLOPS	Processor type ⓘ:	Heron r1	Median readout error: 1.770e-2
	Version:	1.0.16	Median T1: 175.89 us
	Basis gates:	CZ, ID, RZ, SX, X	Median T2: 134.83 us
	Your instance usage:	0 jobs	

Readout errors
because of imperfect measurements.
Most important in shallow circuits.



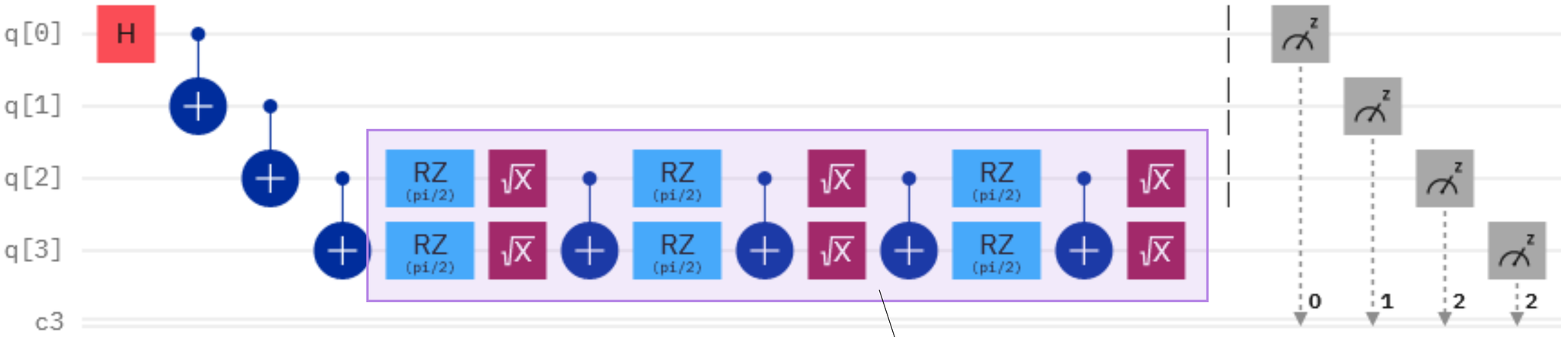
Qubit:

Readout assignment error

Median 1.940e-2

min 3.333e-3 max 4.333e-1

Noise in quantum systems



ibm_torino

OpenQASM 3

Details

133
Qubits

0.8%
EPLG

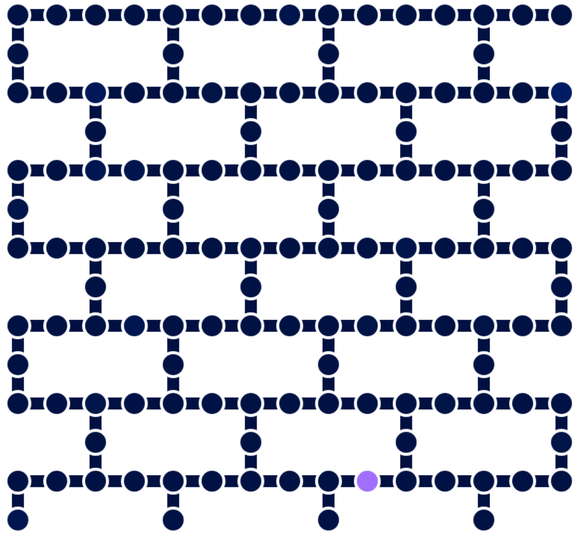
3.8K
CLOPS

Status:	Online
Total pending jobs:	43 jobs
Processor type ⓘ:	Heron r1
Version:	1.0.16
Basis gates:	CZ, ID, RZ, SX, X
Your instance usage:	0 jobs

Median CZ error:	4.384e-3
Median SX error:	3.161e-4
Median readout error:	1.770e-2
Median T1:	175.89 us
Median T2:	134.83 us

Gate errors

Because of imperfect operations on qubits.
(Higher for deep dense circuits with many two-qubit gates).



Qubit: √x (sx) error — Median 3.299e-4 — min 1.335e-4 max 8.469e-2

Connection: CZ error — Median 1.000e+0 — min 1.000e+0 max 1.000e+0

Fighting noise in quantum systems

Fault tolerance

Error correction builds up redundancies that allow us to detect and correct errors when they occur, leading to essentially error-free logical qubits!

Before fault tolerance…

Error suppression techniques transform a circuit during compilation to minimize noise.

- Dynamical decoupling (DD)
- Pauli Twirling (PT)

Before or during execution (typically)

Additional classical resources dominate

Error mitigation techniques reduce circuit errors by modeling the device noise that was present at the time of execution.

- Twirled Readout Error eXtinction (TREX)
- Zero Noise Extrapolation (ZNE)

After or during execution (typically)

Additional quantum resources dominate

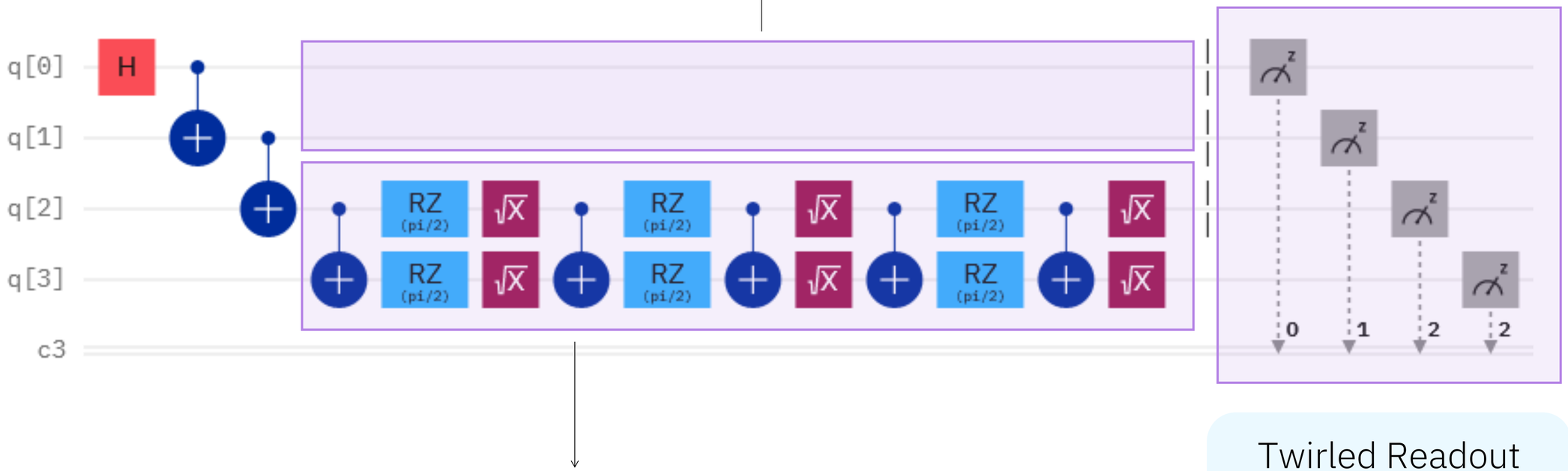
https://www.ibm.com/quantum/blog/quantum-error-suppression-mitigation-correction

Fighting noise in quantum systems

Dynamical decoupling (DD)

Environmental noise

Readout errors



Gate errors

Pauli Twirling (PT)

Zero Noise Extrapolation (ZNE)

Twirled Readout Error eXtinction (TREX)

Fighting noise in quantum systems using Qiskit Runtime

Choose a backend!

```
from qiskit_ibm_runtime import QiskitRuntimeService

service = QiskitRuntimeService()
backend = service.least_busy()
```

✓ 40.8s

Define an object to tweak the options

```
from qiskit_ibm_runtime import SamplerOptions, EstimatorOptions

sampler_options = SamplerOptions(default_shots=1024) #or
estimator_options = EstimatorOptions(default_shots=1024, optimization_level=0, resilience_level=0)
```

✓ 0.0s

We will use Sampler or Estimator V2

```
from qiskit_ibm_runtime import SamplerV2, EstimatorV2

sampler = SamplerV2(backend, options=sampler_options)
estimator = EstimatorV2(backend, options=estimator_options)
```

✓ 0.0s

Sampler options: https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/qiskit_ibm_runtime.options.SamplerOptions

Estimator options: https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/qiskit_ibm_runtime.options.EstimatorOptions

Outline

1 Fighting noise in quantum systems

2 Error suppression techniques

- Dynamical decoupling (DD)
- Pauli Twirling (PT)

3 Error mitigation techniques

- Twirled Readout Error eXtinction (TREX)
- Zero Noise Extrapolation (ZNE)
  - (PEA)

4 Combining techniques and alpha capabilities

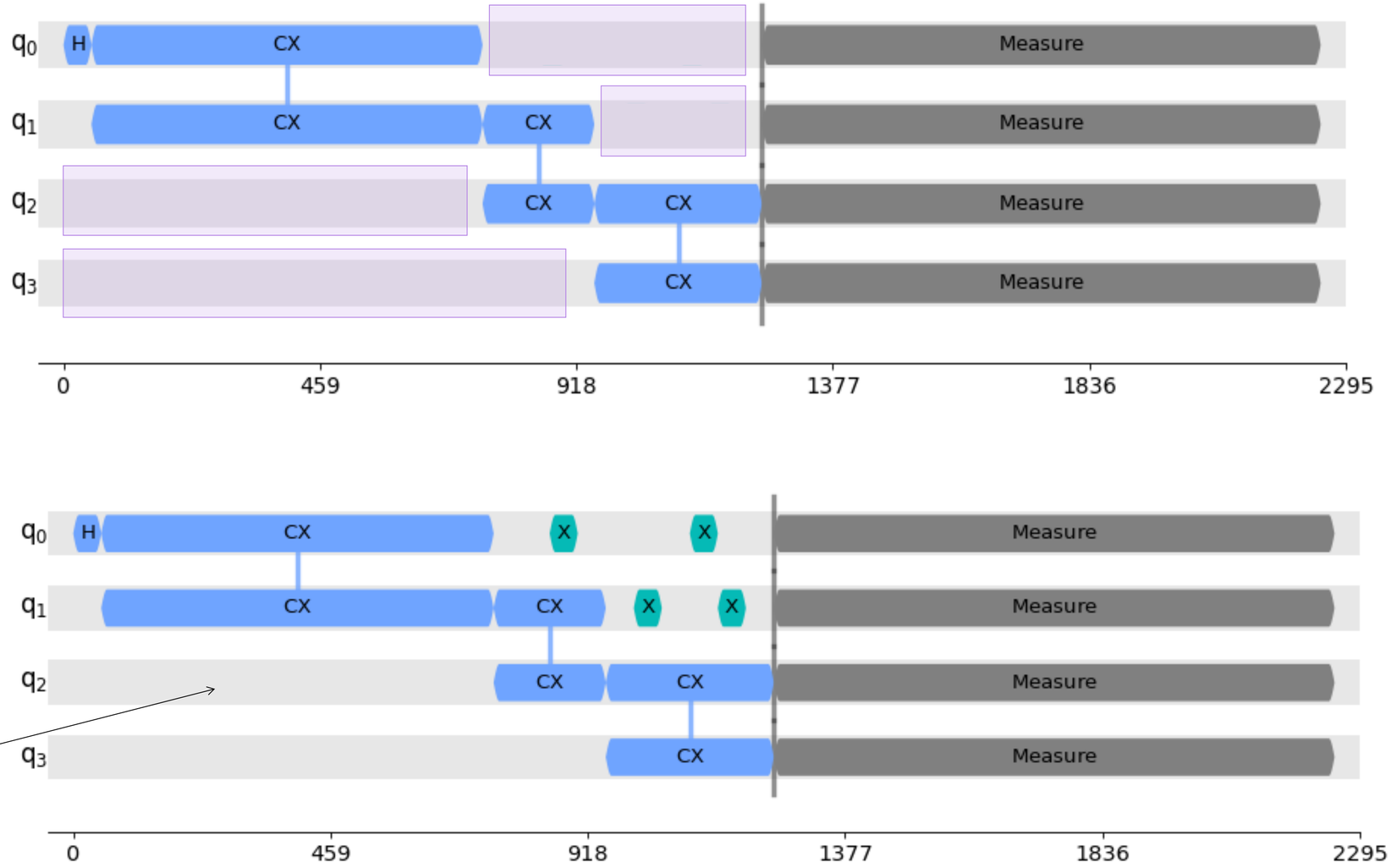
Dynamical decoupling (DD)

(see also https://youtu.be/67jRWQuW3Fk?si=ejmSCg-01NnCxTFk)

Crosstalk can be suppressed by applying gates which add up to the identity.

Such gates will also introduce errors, so there is a balance to be found.

No need to apply gates to qubits which are not initialized.



Dynamical decoupling (DD)

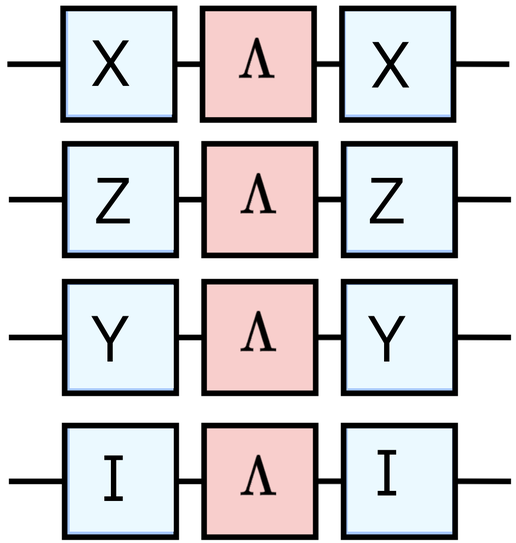
Options	Sub-options	Sub-sub-options	Choices	Default
dynamical_decoupling	enable		`True`/`False`	`False`
	sequence_type		`'XX'`/`'XpXm'`/`'XY4'`	`'XX'`
	extra_slack_distribution		`'middle'`/`'edges'`	`'middle'`
	scheduling_method		`'asap'`/`'alap'`	`'alap'`

```
from qiskit_ibm_runtime import SamplerOptions, EstimatorOptions

options = SamplerOptions(default_shots=1024)  # or...
options = EstimatorOptions(default_shots=1024, optimization_level=0, resilience_level=0)

## Configure Dynamical Decoupling
options.dynamical_decoupling.enable = True
options.dynamical_decoupling.sequence_type = 'XX'
options.dynamical_decoupling.extra_slack_distribution = 'middle'
options.dynamical_decoupling.scheduling_method = 'alap'
```

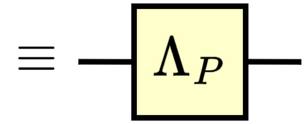
https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/qiskit_ibm_runtime.options.DynamicalDecouplingOptions

Pauli Twirling (PT)

Arbitrary noise channels can be converted into Pauli noise.



Average over instances with equal probability

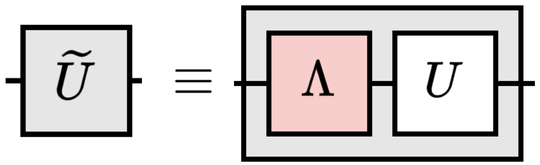


Pauli noise accumulates linearly, in contrast to coherent noise, which accumulates quadratically!

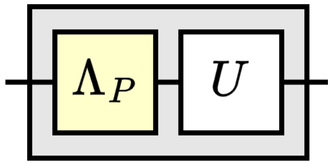
Pauli Twirling (PT)

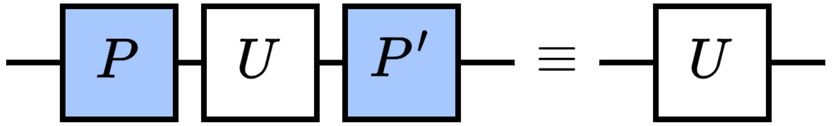
We can consider noisy gates to be associated to a noisy channel Λ.

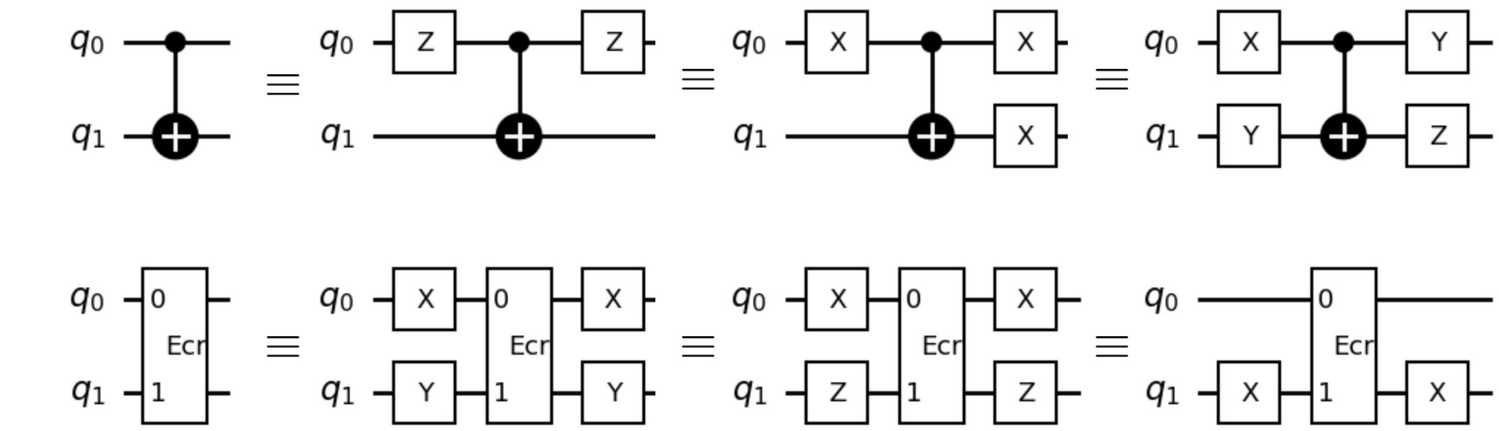
Arbitrary quantum channel



Pauli channel







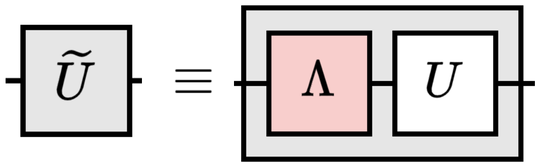
Set of all possible combinations for a given gate.

(for Clifford gates 4x4=16 possibilities)

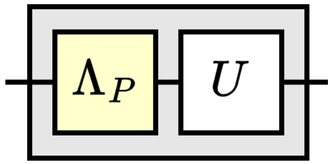
Pauli Twirling (PT)

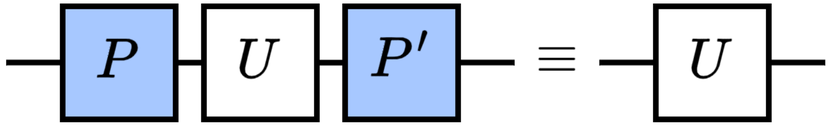
We can consider noisy gates to be associated to a noisy channel Λ.

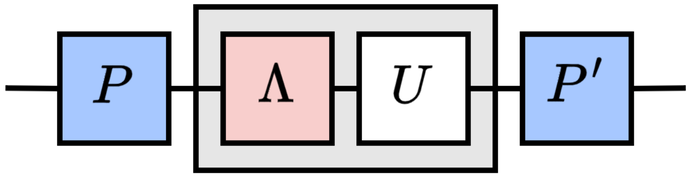
Arbitrary quantum channel



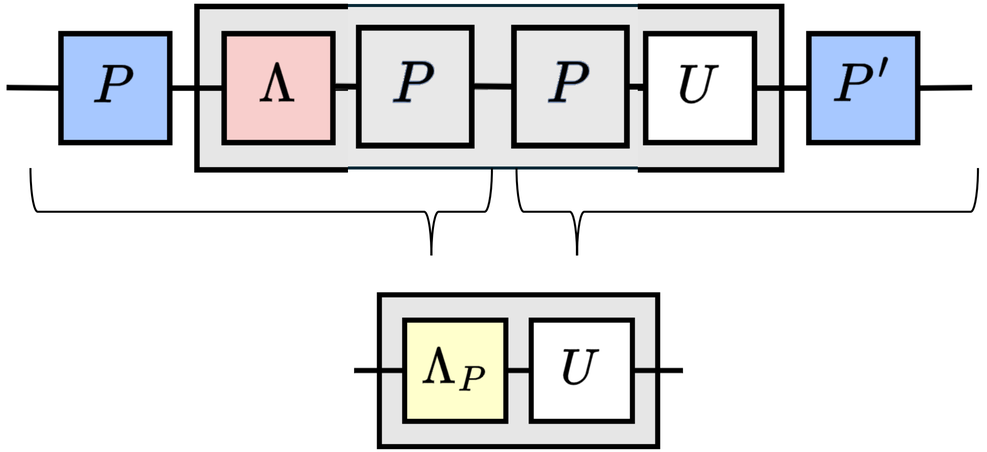
Pauli channel







Average over random instances from the set of possible combinations

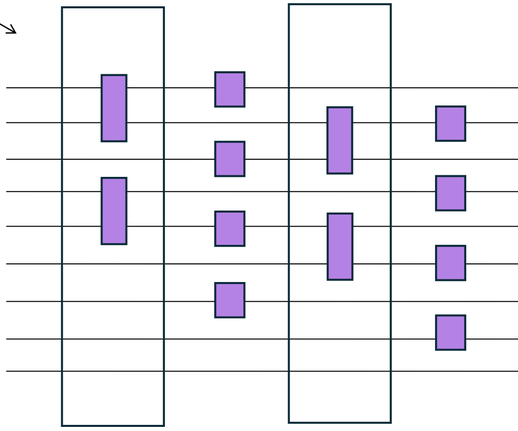


Pauli Twirling (PT)

Options	Sub-options	Sub-sub-options	Choices	Default
twirling	enable_gates		`True`/`False`	`False`
	enable_measure		`True`/`False`	`True`
	num_randomizations			`'auto'`
	shots_per_randomization			`'auto'`
	strategy		`'active'`/ `'active-circuit'`/ `'active-accum'`/ `'all'`	`'active-accum'`

(later)

```
from qiskit_ibm_runtime import SamplerOptions, EstimatorOptions

options = SamplerOptions(default_shots=1024)  # or...
options = EstimatorOptions(default_shots=1024, optimization_level=0, resilience_level=0)

## Configure Twirling
options.twirling.enable_gates = True
options.twirling.enable_measure = False
options.twirling.num_randomizations = 'auto'
options.twirling.shots_per_randomization = 'auto'
options.twirling.strategy = 'active-accum'
```



https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/qiskit_ibm_runtime.options.TwirlingOptions

Outline

1 Fighting noise in quantum systems

2 Error suppression techniques

- Dynamical decoupling (DD)
- Pauli Twirling (PT)

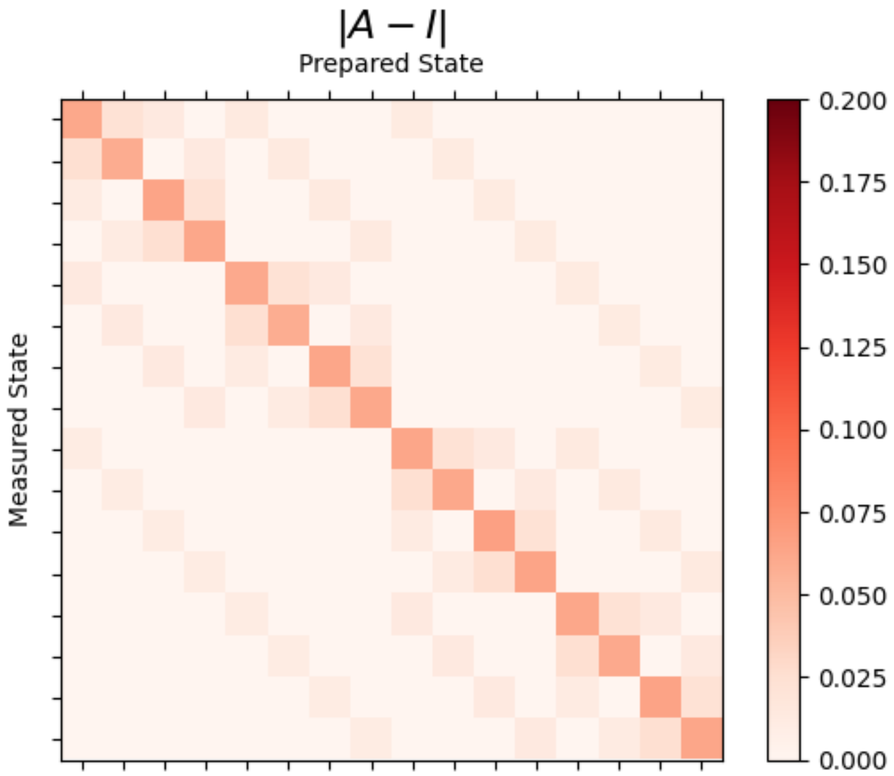
3 Error mitigation techniques

- Twirled Readout Error eXtinction (TREX)
- Zero Noise Extrapolation (ZNE)
  - (PEA)

Only valid for Estimator!

4 Combining techniques and alpha capabilities

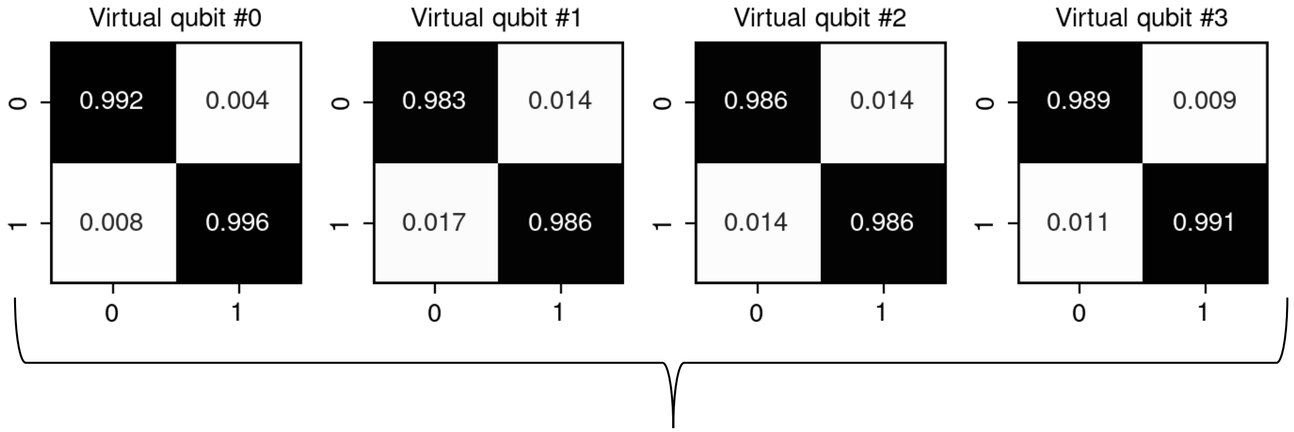
Twirled Readout Error eXtinction (TREX)



Readouts errors cause the wrong states to be measured.

This is depicted in the readout-error transfer matrix.

In scenarios when the noise is not correlated between qubits, measurement errors can be measured per qubit and the full transfer matrix is then reconstructed as a tensor product.



$$2^N \times 2^N$$

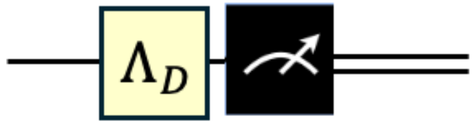
The inverse of the transfer matrix can be used for error mitigation, but it cannot be obtained efficiently in general!

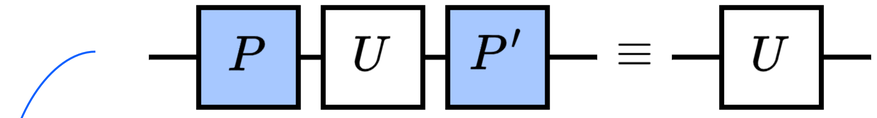
Twirled Readout Error eXtinction (TREX)

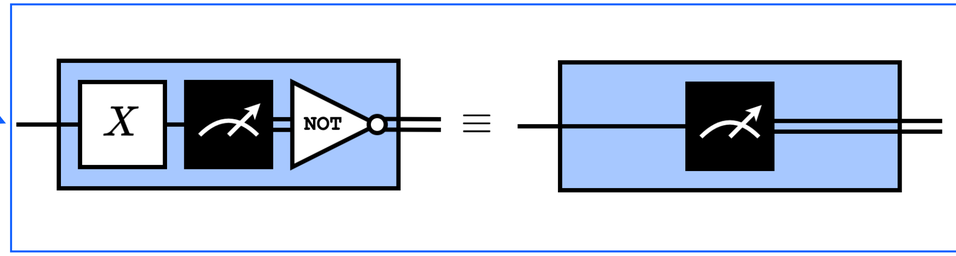
Via **measurement twirling we can diagonalize the** readout-error transfer matrix.

Arbitrary transfer matrix

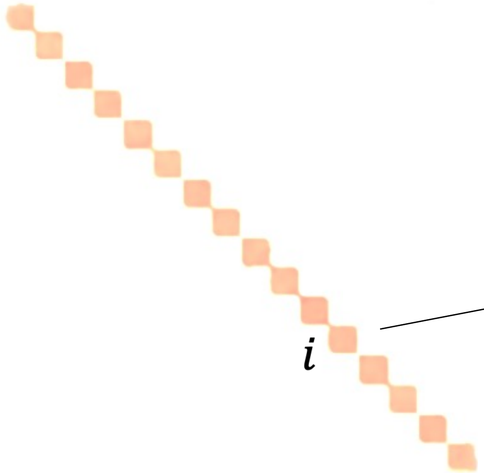
Diagonal transfer matrix







Average over random bitflips that are undone in classical postprocessing.



Then, we need to run calibration circuits to get the inverse of the (diagonal) readout error transfer matrix.

$$< O_i > = \frac{< \widetilde{O_i} >}{E_i}$$

Only valid for expectation values!

Twirled Readout Error eXtinction (TREX)

Options	Sub-options	Sub-sub-options	Choices	Default
resilience	measure_mitigation		`True`/`False`	`True`
	measure_noise_learning	num_randomizations		`32`
		shots_per_randomization		`'auto'`

```
from qiskit_ibm_runtime import EstimatorOptions

options = EstimatorOptions(default_shots=1024, optimization_level=0, resilience_level=0)

## Configure TREX
options.resilience.measure_mitigation = True
options.resilience.measure_noise_learning.num_randomizations = 32
options.resilience.measure_noise_learning.shots_per_randomization = 'auto'

options.twirling.enable_measure = True  # Automatically set by TREX
```

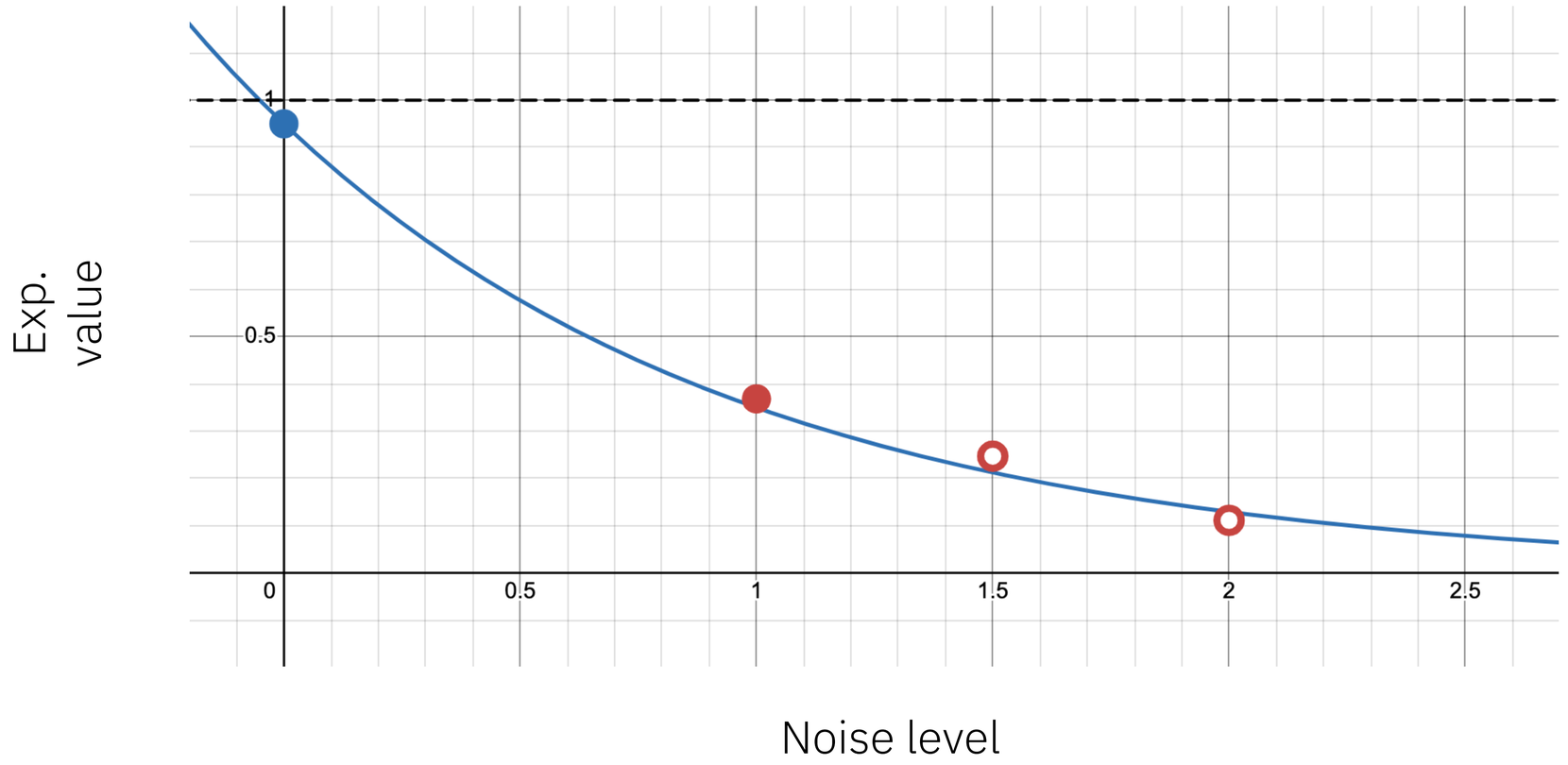
https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/qiskit_ibm_runtime.options.ResilienceOptionsV2

Zero Noise Extrapolation (ZNE)

Measures the effects of amplified noise to infer what the results would look like in the absence of noise.

1. Noise amplification:
   the original circuit unitary is executed at different levels of noise.

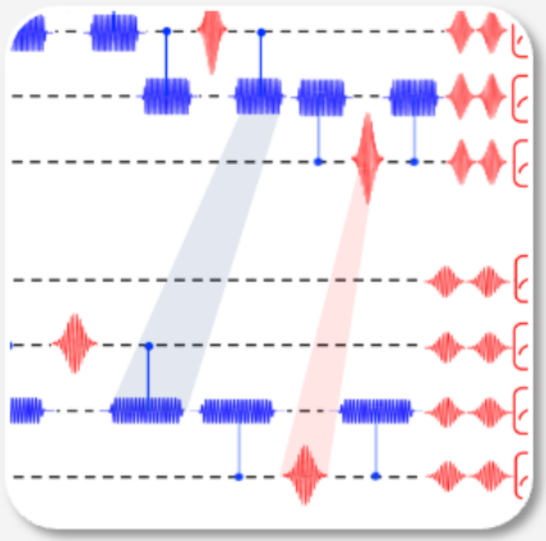
2. Extrapolation:
   the zero-noise limit is inferred from the noisy expectation-value results.



Zero Noise Extrapolation (ZNE)

Step 1: Noise amplification

Pulse stretching	Gate folding	Probabilistic error amplification
Scale pulse duration via calibration	Repeat gates in identity cycles $U \mapsto U(U^{-1}U)^{\lambda-1}/2$	Add noise via sampling Pauli channels



Kandala et al. Nature (2019)

It’s an analogue technique.

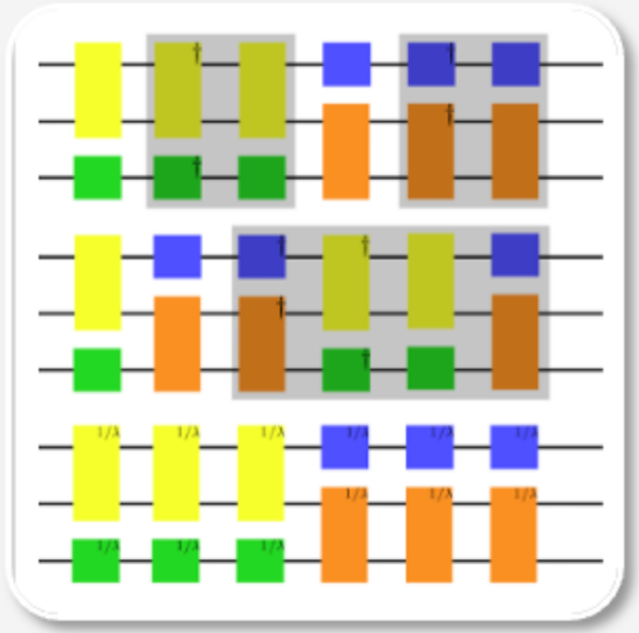
It assumes noise is proportional to pulse duration.

It requires costly pulse level calibration of the hardware.

Zero Noise Extrapolation (ZNE)

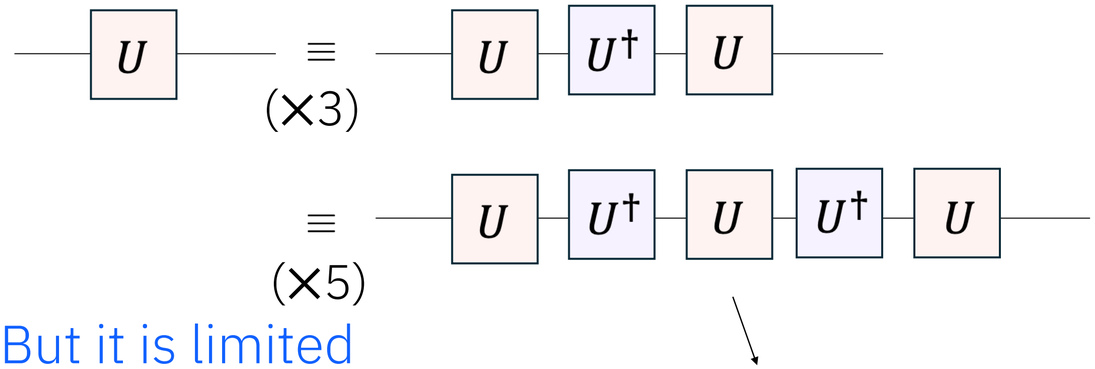
Step 1: Noise amplification

Pulse stretching	Gate folding	Probabilistic error amplification
Scale pulse duration via calibration	Repeat gates in identity cycles $U \mapsto U(U^{-1}U)^{\lambda-1}/2$	Add noise via sampling Pauli channels



Shultz et al. PRA (2022)

It is a heuristic approach but offers a good trade-off between result quality and resource requirements.



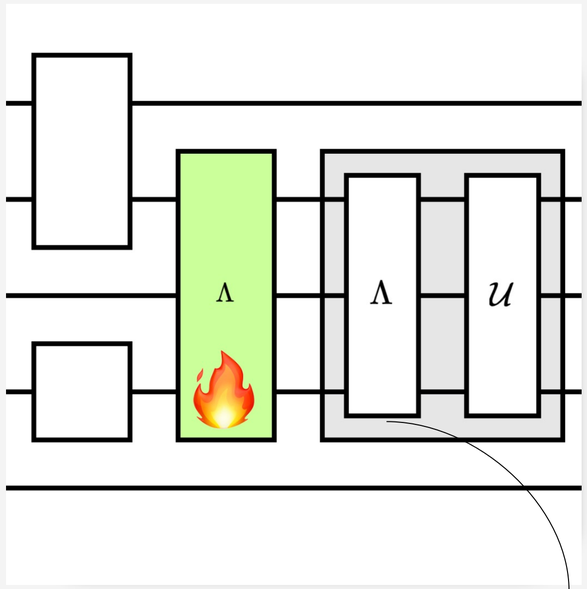
But it is limited by circuit depth!

especially powerful when $U \equiv U^\dagger$

Zero Noise Extrapolation (ZNE)

Step 1: Noise amplification

Pulse stretching	Gate folding	Probabilistic error amplification
Scale pulse duration via calibration	Repeat gates in identity cycles $U \mapsto U(U^{-1}U)^{\lambda-1}/2$	Add noise via sampling Pauli channels



Li & Benjamin PRX (2017)

It has general applicability and strong theoretical backing.

Used in the utility paper
Y. Kim et al. Nature (2023).

But it requires learning circuit-specific noise.

Idea: we want to learn the noise channel of certain layers of gates and reproduce it.

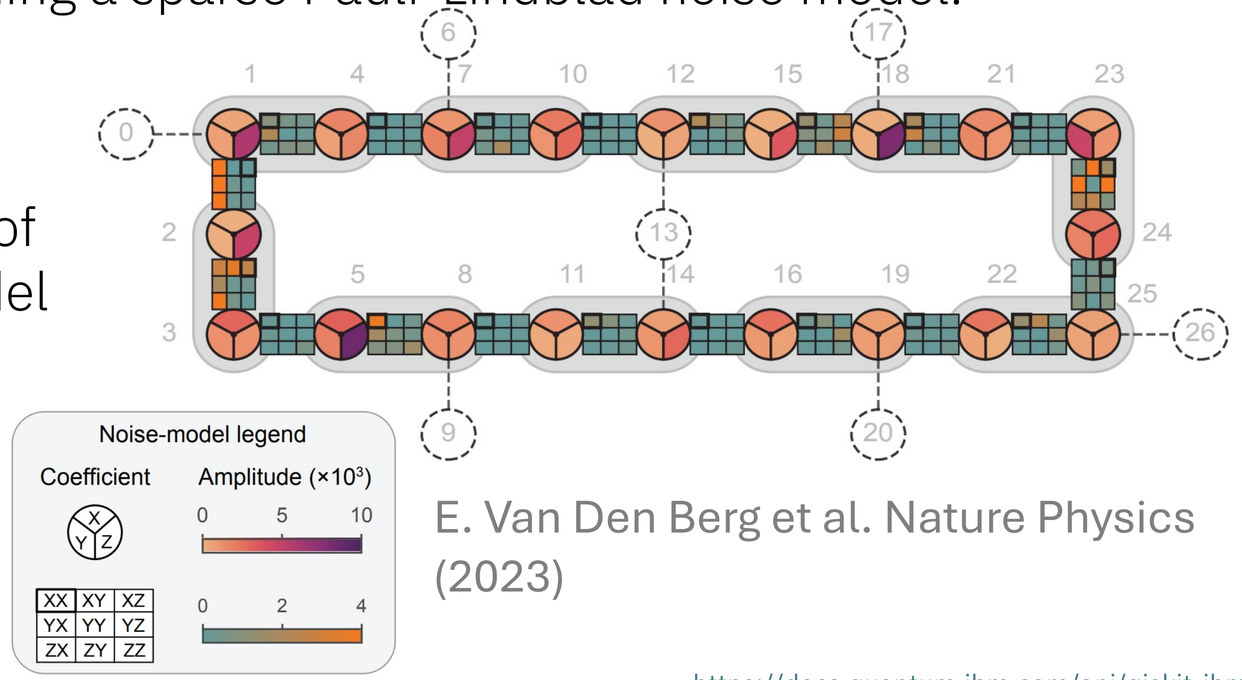
Zero Noise Extrapolation (ZNE)

Step 1: Noise amplification

Pulse stretching	Gate folding	Probabilistic error amplification
Scale pulse duration via calibration	Repeat gates in identity cycles $U \mapsto U(U^{-1}U)^{\lambda-1}/2$	Add noise via sampling Pauli channels

- To learn the noise, we can use the noise learner to characterize the noise affecting the 2Q gate layers in a given circuit.
- It learns the noise assuming a sparse Pauli-Lindblad noise model.
- The parameters of the resulting model scale linearly with the number of qubits, therefore the model is efficiently represented and easy to learn.

Caveat: The learning requires more circuit executions and therefore adds overhead!



E. Van Den Berg et al. Nature Physics (2023)

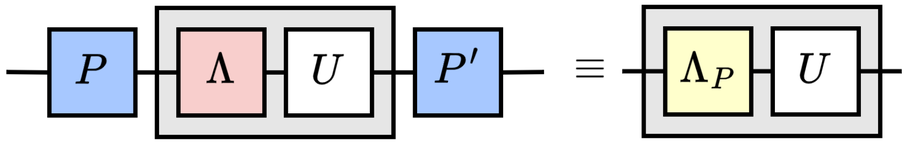
https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/0.26/qiskit_ibm_runtime.noise_learner.NoiseLearner

Zero Noise Extrapolation (ZNE)

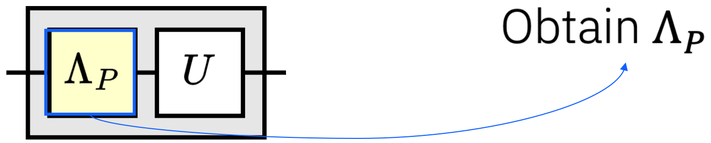
Step 1: Noise amplification

Pulse stretching	Gate folding	Probabilistic error amplification
Scale pulse duration via calibration	Repeat gates in identity cycles $U \mapsto U(U^{-1}U)^{\lambda-1}/2$	Add noise via sampling Pauli channels

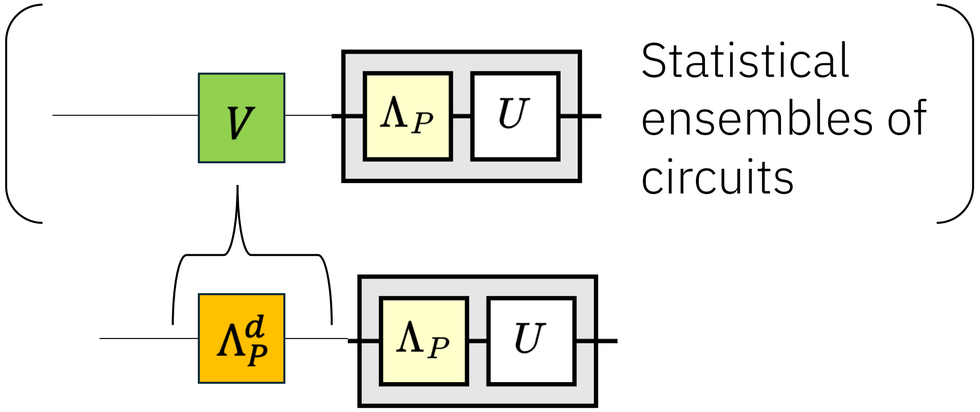
Step 1. Use Pauli Twirling to transform the noise channels to Pauli noise, which is easier to learn.



Step 2. Noise learning.



Step 3. Noise injection using random unitaries.



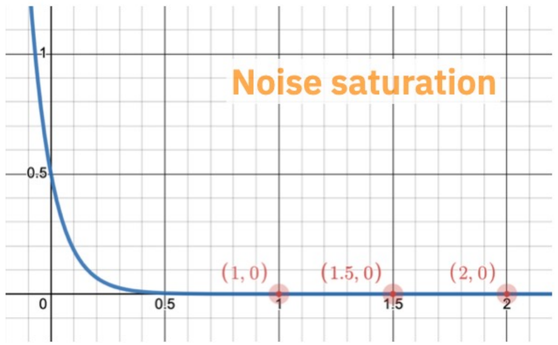
Zero Noise Extrapolation (ZNE)

Step 2. Extrapolation

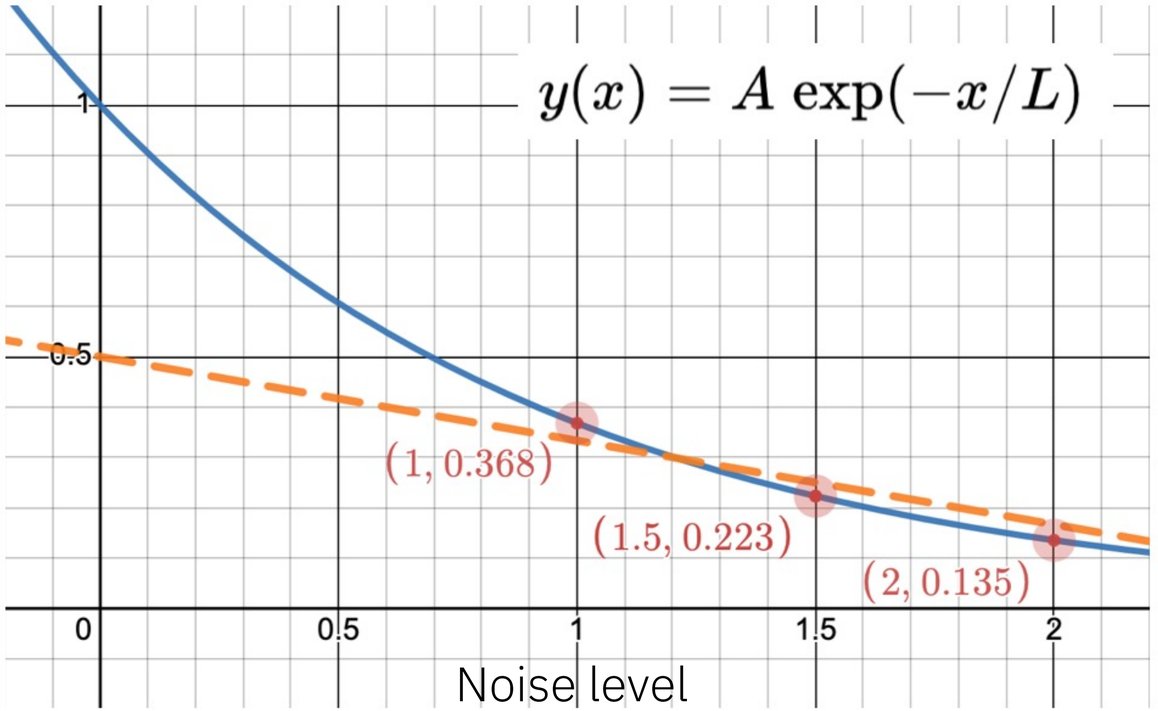
Theoretical/experimental results predict exponential decay in observed expectation values

Exponential extrapolation mitigates aggressively but is unstable because we don't know the scale.

$A = \pm 0.5 \; ; \; L = 0.1$



$A = 1.0 \; ; \; L = 1.0$



Polynomial extrapolation is stable but mitigates worse, since it retains the scale of the noisy data.

Zero Noise Extrapolation (ZNE)

Options	Sub-options	Sub-sub-options	Choices	Default
resilience	zne_mitigation		`True`/`False`	`False`
	zne	noise_factors		`(1, 1.5, 2) for PEA, and (1, 3, 5) otherwise`
		amplifier	`'gate_folding'`/ `'gate_folding_front'`/ `'gate_folding_back'`/ `'pea'`	`gate_folding`
		extrapolator	`'exponential'`/ `'linear'`/ `'double_exponential'`/ `'polynomial_degree_(1 <= k <= 7)'`	(`'exponential'`, `'linear'`)

```python
from qiskit_ibm_runtime import EstimatorOptions

options = EstimatorOptions(default_shots=1024, optimization_level=0, resilience_level=0)

## Configure ZNE
options.resilience.zne_mitigation = True
options.resilience.zne.noise_factors = (1, 3, 5)  [Can have fractional noise factors, e.g. (1, 1.5, 2)]
options.resilience.zne.amplifier = 'gate_folding'
options.resilience.zne.extrapolator = ('exponential', 'linear')
```

Choosing the right noise factors and extrapolator is tricky!

https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/qiskit_ibm_runtime.options.ResilienceOptionsV2

Zero Noise Extrapolation (ZNE)

PEA – Noise learner

```
from qiskit_ibm_runtime import EstimatorOptions

options = EstimatorOptions(default_shots=1024, optimization_level=0, resilience_level=0)

## Configure ZNE
options.resilience.zne_mitigation = True
options.resilience.zne.noise_factors = (1, 3, 5)
options.resilience.zne.amplifier = 'pea'
options.resilience.zne.extrapolator = ('exponential', 'linear')
```

```
options.resilience.layer_noise_learning.max_layers_to_learn = 4
options.resilience.layer_noise_learning.num_randomizations = 32
options.resilience.layer_noise_learning.shots_per_randomization = 128
options.resilience.layer_noise_learning.layer_pair_depths = (0, 1, 2, 4, 16, 32)
```

https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/qiskit_ibm_runtime.options.LayerNoiseLearningOptions

Outline

1 Fighting noise in quantum systems

2 Error suppression techniques

- Dynamical decoupling (DD)
- Pauli Twirling (PT)

3 Error mitigation techniques

- Twirled Readout Error eXtinction (TREX)
- Zero Noise Extrapolation (ZNE)
  - (PEA)
- Others: PEC

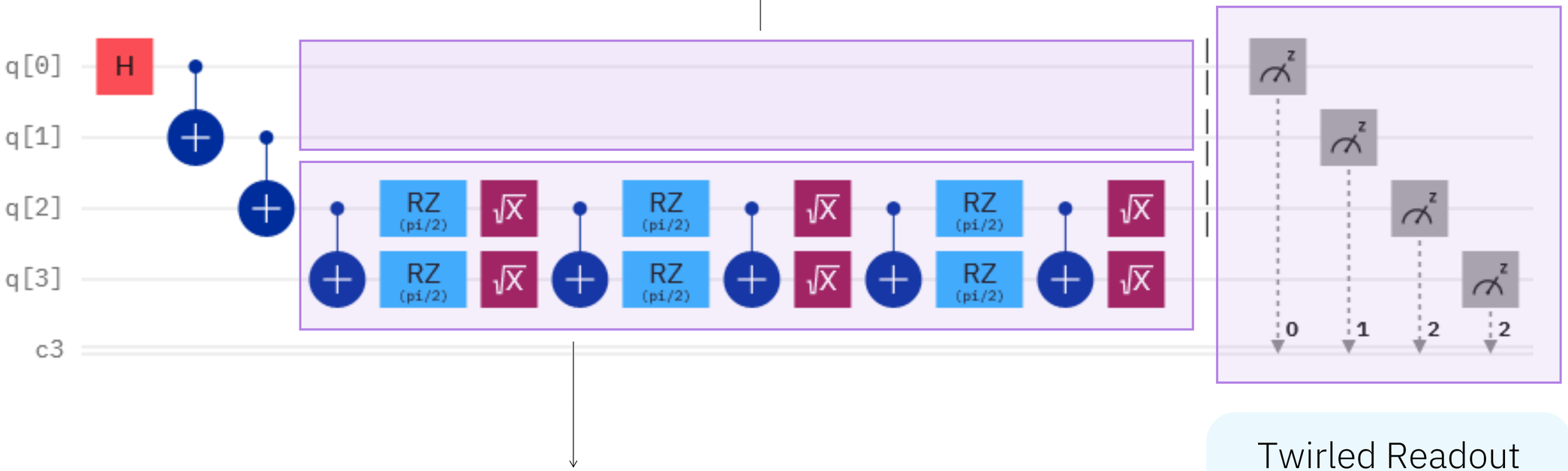
4 Combining techniques

Combining techniques

(3) Dynamical decoupling (DD)

Readout errors

Environmental noise



Gate errors

(1) Pauli Twirling (PT)

(2) Zero Noise Extrapolation (ZNE)

Twirled Readout Error eXtinction (TREX)

Combining techniques

```
from qiskit_ibm_runtime import QiskitRuntimeService, EstimatorV2, EstimatorOptions

options = EstimatorOptions(default_shots=1024, optimization_level=0, resilience_level=0)

## Configure Dynamical Decoupling
options.dynamical_decoupling.enable = True
options.dynamical_decoupling.sequence_type = 'XX'
options.dynamical_decoupling.extra_slack_distribution = 'middle'
options.dynamical_decoupling.scheduling_method = 'alap'

## Configure Twirling
options.twirling.enable_gates = True
options.twirling.enable_measure = True  # Needed for TREX
options.twirling.num_randomizations = 'auto'
options.twirling.shots_per_randomization = 'auto'
options.twirling.strategy = 'active-accum'

## Configure TREX
options.resilience.measure_mitigation = True
options.resilience.measure_noise_learning.num_randomizations = 32
options.resilience.measure_noise_learning.shots_per_randomization = 'auto'

## Configure ZNE
options.resilience.zne_mitigation = True
options.resilience.zne.noise_factors = (1, 3, 5)
options.resilience.zne.extrapolator = 'exponential'

service = QiskitRuntimeService()
backend = service.least_busy()
estimator = EstimatorV2(backend, options=options)
```

Combining techniques

Useful summary of all techniques:
https://docs.quantum.ibm.com/guides/error-mitigation-and-suppression-techniques

```python
from qiskit_ibm_runtime import EstimatorV2

estimator = EstimatorV2(backend=backend)
estimator.options.resilience_level = 2
job = estimator.run([(transpiled_circuit, transpiled_observable)], precision=0.01)
print(f">>> Job ID: {job.job_id()}")
print(f">>> Job Status: {job.status()}")
```

✓ 2.1s Python

Resilience Level	Definition	Technique
0	No mitigation	None
1 [Default]	Minimal mitigation costs: Mitigate error associated with readout errors	Twirled Readout Error eXtinction (TREX) measurement twirling
2	Medium mitigation costs. Typically reduces bias in estimators, but is not guaranteed to be zero-bias.	Level 1 + Zero Noise Extrapolation (ZNE) and gate twirling

resilience level is only available for Estimator!

Tutorial:

Combine error mitigation options with the Estimator primitive

https://learning.quantum.ibm.com/tutorial/combine-error-mitigation-options-with-the-estimator-primitive