


Error Mitigation with Mitiq

Nate Stemen

Sep 18, 2023







Mitiq is a Python toolkit for implementing error mitigation techniques on quantum computers.


Current quantum computers are noisy due to interactions with the environment, imperfect gate applications, state preparation and measurement errors, etc. Error mitigation seeks to reduce these effects at the software level by compiling quantum programs in clever ways.


Want to know more? Check out our [documentation](#) and chat with us on [Discord](#).


README.md





 build passing


 docs passing

 codecov 98%


 pypi package 0.29.0


 arXiv 2009.04417

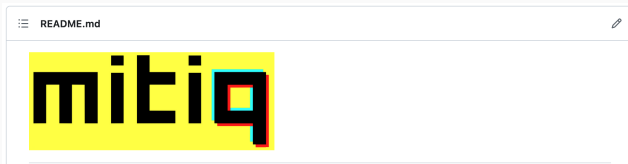
 Downloads 100k

 GitHub

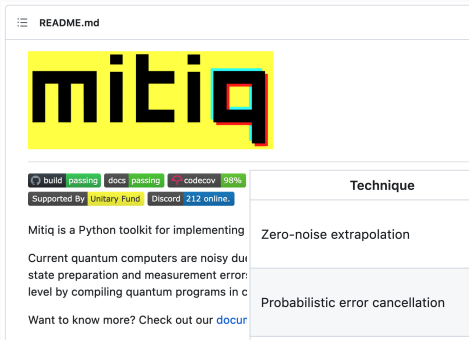
Supported By

 Unitary Fund

 Discord 212 online



Technique	Documentation	Mitq module	Paper Reference(s)
Zero-noise extrapolation	ZNE	mitiq.zne	1611.09301 1612.02058 1805.04492
Probabilistic error cancellation	PEC	mitiq.pec	1612.02058 1712.09271 1905.10135
(Variable-noise) Clifford data regression	CDR	mitiq.cdr	2005.10189 2011.01157
Digital dynamical decoupling	DDD	mitiq.ddd	9803057 1807.08768
Readout-error mitigation	REM	mitiq.rem	1907.08518 2006.14044



```
import cirq

+ import mitiq

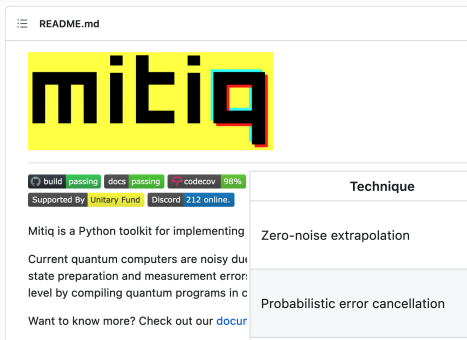
qubit = cirq.LineQubit(1)
circuit = cirq.Circuit(cirq.X(qubit) for _ in range(100))

- expval = execute(circuit)
+ expval = mitiq.zne.execute_with_zne(circuit, execute)

print(f"Error: {1 - expval:.3}")

- # Error: 0.244
+ # Error: 0.058
```

Technique			
Zero-noise extrapolation	<pre>print(f"Error: {1 - expval:.3}") - # Error: 0.244 + # Error: 0.058</pre>		
Probabilistic error cancellation	PEC	<code>mitiq.pec</code>	1712.09271 1905.10135
(Variable-noise) Clifford data regression	CDR	<code>mitiq.cdr</code>	2005.10189 2011.01157
Digital dynamical decoupling	DDD	<code>mitiq.ddd</code>	9803057 1807.08768
Readout-error mitigation	REM	<code>mitiq.rem</code>	1907.08518 2006.14044



```
import cirq
+ import mitiq

qubit = cirq.LineQubit(1)
circuit = cirq.Circuit(cirq.X(qubit) for _ in range(100))

- expval = execute(circuit)
+ expval = mitiq.zne.execute_with_zne(circuit, execute)

print(f"Error: {1 - expval:.3}")
- # Error: 0.244
+ # Error: 0.058
```

Technique			
Zero-noise extrapolation	<pre>print(f"Error: {1 - expval:.3}")</pre>		
	<pre>- # Error: 0.244</pre>		
Probabilistic error cancellation	<pre>+ # Error: 0.058</pre>		
	PEC	<code>mitiq.pec</code>	1712.09271 1905.10135
(Variable-noise) Clifford data regression	CDR	<code>mitiq.cdr</code>	2005.10189 2011.01157
Digital dynamical decoupling	DDD	<code>mitiq.ddd</code>	9803057 1807.08768
Readout-error mitigation	REM	<code>mitiq.rem</code>	1907.08518 2006.14044



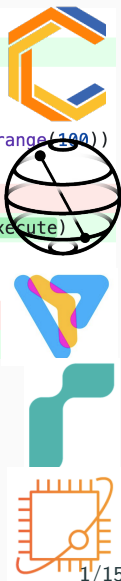
```
import cirq
+ import mitiq

qubit = cirq.LineQubit(1)
circuit = cirq.Circuit(cirq.X(qubit) for _ in range(100))

- expval = execute(circuit)
+ expval = mitiq.zne.execute_with_zne(circuit, execute)

print(f"Error: {1 - expval:.3}")
- # Error: 0.244
+ # Error: 0.058
```

Technique			
Zero-noise extrapolation	<pre>print(f"Error: {1 - expval:.3}") - # Error: 0.244 + # Error: 0.058</pre>		
Probabilistic error cancellation	PEC	<code>mitiq.pec</code>	1712.09271 1905.10135
(Variable-noise) Clifford data regression	CDR	<code>mitiq.cdr</code>	2005.10189 2011.01157
Digital dynamical decoupling	DDD	<code>mitiq.ddd</code>	9803057 1807.08768
Readout-error mitigation	REM	<code>mitiq.rem</code>	1907.08518 2006.14044



Follow along!



<https://github.com/unitaryfund/mitiq-tutorial>

1. Who has written a quantum program before?

1. Who has written a quantum program before?
2. Who has run a quantum program on hardware before?

1. Who has written a quantum program before?
2. Who has run a quantum program on hardware before?
3. Who has used error mitigation?

1. Who has written a quantum program before?
2. Who has run a quantum program on hardware before?
3. Who has used error mitigation?
4. Who has used Mitiq?

1. Understand context, and general ideas of quantum error mitigation (QEM).
2. Understand main ideas of ZNE, PEC, and DDD along with pros and cons of each technique.
3. Ability to use Mitiq to apply these techniques in a quantum pipeline.

What is Quantum Error Mitigation?

Quantum Error Mitigation

The acceptance that available quantum devices are noisy. . . maybe very much so.
But we still want to use them!

What is Quantum Error Mitigation?

Quantum Error Mitigation

The acceptance that available quantum devices are noisy. . . maybe very much so.
But we still want to use them!

- (In)coherent noise
- SPAM errors

What is Quantum Error Mitigation?

Quantum Error Mitigation

The acceptance that available quantum devices are noisy. . . maybe very much so.
But we still want to use them!

- (In)coherent noise
- SPAM errors
- Crosstalk

What is Quantum Error Mitigation?

Quantum Error Mitigation

The acceptance that available quantum devices are noisy. . . maybe very much so.
But we still want to use them!

- (In)coherent noise
- SPAM errors
- Crosstalk
- Calibration errors

What is Quantum Error Mitigation?

Quantum Error Mitigation

The acceptance that available quantum devices are noisy. . . maybe very much so.
But we still want to use them!

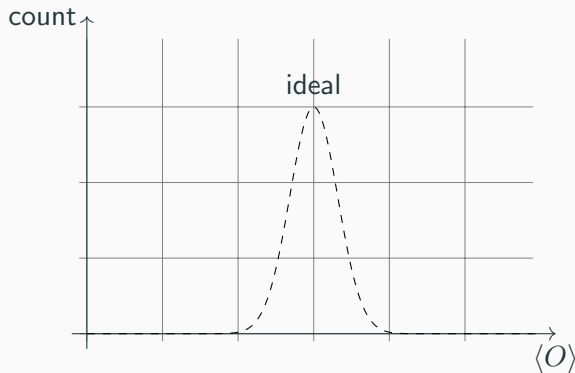
- (In)coherent noise
- SPAM errors
- Crosstalk
- Calibration errors
- . . .

What is Quantum Error Mitigation?

Quantum Error Mitigation

The acceptance that available quantum devices are noisy... maybe very much so.
But we still want to use them!

- (In)coherent noise
- SPAM errors
- Crosstalk
- Calibration errors
- ...

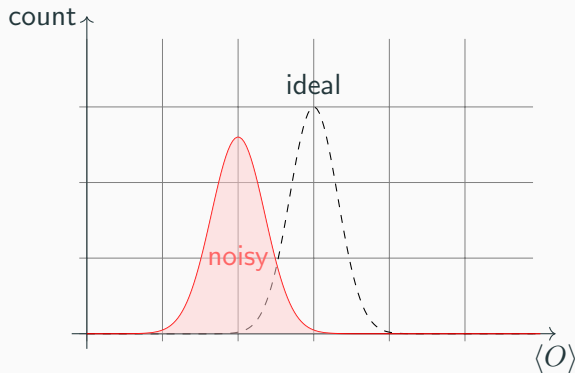


What is Quantum Error Mitigation?

Quantum Error Mitigation

The acceptance that available quantum devices are noisy... maybe very much so.
But we still want to use them!

- (In)coherent noise
- SPAM errors
- Crosstalk
- Calibration errors
- ...

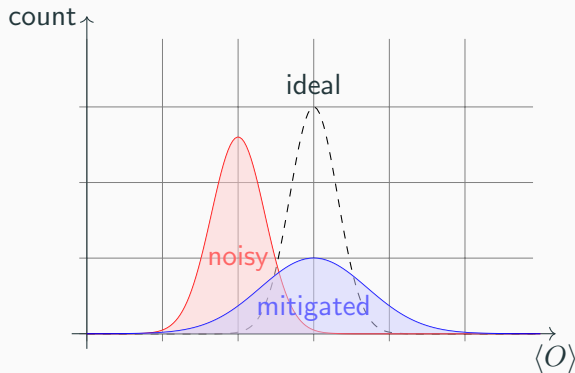


What is Quantum Error Mitigation?

Quantum Error Mitigation

The acceptance that available quantum devices are noisy. . . maybe very much so.
But we still want to use them!

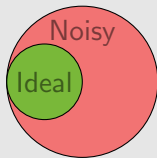
- (In)coherent noise
- SPAM errors
- Crosstalk
- Calibration errors
- . . .



Zero-Noise Extrapolation

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$

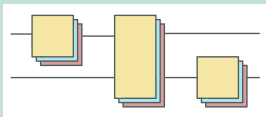
Symmetry-based techniques



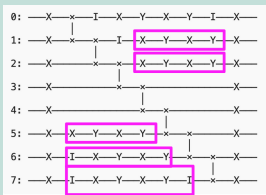
$$M|\psi\rangle = |\psi\rangle$$

$$\rho = \frac{M\rho M}{\text{tr}(M\rho)}$$

Probabilistic Error Cancellation

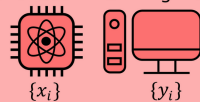


Dynamical Decoupling



Learning- based methods

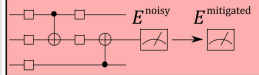
Generate Training Data



Learn To Correct



Predict



What about error correction?

Scheme for reducing decoherence in quantum computer memory

Peter W. Shor*

AT&T Bell Laboratories, Room 2D-149, 600 Mountain Avenue, Murray Hill, New Jersey 07974

(Received 17 May 1995)

Recently, it was realized that use of the properties of quantum mechanics might speed up certain computations dramatically. Interest has since been growing in the area of quantum computation. One of the main difficulties of quantum computation is that decoherence destroys the information in a superposition of states contained in a quantum computer, thus making long computations impossible. It is shown how to reduce the effects of decoherence for information stored in quantum memory, assuming that the decoherence process acts independently on each of the bits stored in memory. This involves the use of a quantum analog of error-correcting codes.

What about error correction?

Scheme for reducing decoherence in quantum computer memory

Peter W. Shor*

AT&T Bell Laboratories, Room 2D-149, 600 Mountain Avenue, Murray Hill, New Jersey 07974

(Received 17 May 1995)

Recently, it was realized that use of the properties of quantum mechanics might speed up certain computations dramatically. Interest has since been growing in the area of quantum computation. One of the main difficulties of quantum computation is that decoherence destroys the information in a superposition of states contained in a quantum computer, thus making long computations impossible. It is shown how to reduce the effects of decoherence for information stored in quantum memory, assuming that the decoherence process acts independently on each of the bits stored in memory. This involves the use of a quantum analog of error-correcting codes.

Error Correction

- Encode logical qubits into many physical qubits
- Intermediate measurements produce syndromes
- Use syndromes to correct errors

What about error correction?

Scheme for reducing decoherence in quantum computer memory

Peter W. Shor*

AT&T Bell Laboratories, Room 2D-149, 600 Mountain Avenue, Murray Hill, New Jersey 07974

(Received 17 May 1995)

Recently, it was realized that use of the properties of quantum mechanics might speed up certain computations dramatically. Interest has since been growing in the area of quantum computation. One of the main difficulties of quantum computation is that decoherence destroys the information in a superposition of states contained in a quantum computer, thus making long computations impossible. It is shown how to reduce the effects of decoherence for information stored in quantum memory, assuming that the decoherence process acts independently on each of the bits stored in memory. This involves the use of a quantum analog of error-correcting codes.

Error Correction

- Encode logical qubits into many physical qubits
- Intermediate measurements produce syndromes
- Use syndromes to correct errors

Error Mitigation

- Perform multiple and different noisy computations
- Collect results
- Infer ideal expectation values

What about error correction?

Scheme for reducing decoherence in quantum computer memory

Peter W. Shor*

AT&T Bell Laboratories, Room 2D-149, 600 Mountain Avenue, Murray Hill, New Jersey 07974

(Received 17 May 1995)

Recently, it was realized that use of the properties of quantum mechanics might speed up certain computations dramatically. Interest has since been growing in the area of quantum computation. One of the main difficulties of quantum computation is that decoherence destroys the information in a superposition of states contained in a quantum computer, thus making long computations impossible. It is shown how to reduce the effects of decoherence for information stored in quantum memory, assuming that the decoherence process acts independently on each of the bits stored in memory. This involves the use of a quantum analog of error-correcting codes.

Error Correction

- Encode logical qubits into many physical qubits
- Interactions between qubits produce syndromes
- Use syndromes to correct errors

Scalable, but unfeasible

Error Mitigation

- Perform multiple and different noisy computations
- Collect results
- Infer ideal expectation values

What about error correction?

Scheme for reducing decoherence in quantum computer memory

Peter W. Shor*

AT&T Bell Laboratories, Room 2D-149, 600 Mountain Avenue, Murray Hill, New Jersey 07974

(Received 17 May 1995)

Recently, it was realized that use of the properties of quantum mechanics might speed up certain computations dramatically. Interest has since been growing in the area of quantum computation. One of the main difficulties of quantum computation is that decoherence destroys the information in a superposition of states contained in a quantum computer, thus making long computations impossible. It is shown how to reduce the effects of decoherence for information stored in quantum memory, assuming that the decoherence process acts independently on each of the bits stored in memory. This involves the use of a quantum analog of error-correcting codes.

Error Correction

- Encode logical qubits into many physical qubits
- Interactions between qubits produce syndromes
- Use syndromes to correct errors

Scalable, but unfeasible

Error Mitigation

- Perform multiple and different noisy computations
- Collect results
- Infer expectation values

Unscalable*, but feasible

Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

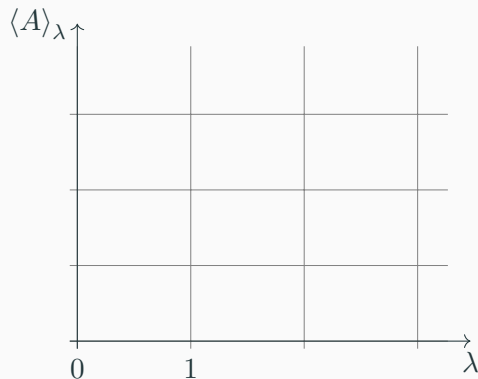
$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$

Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$

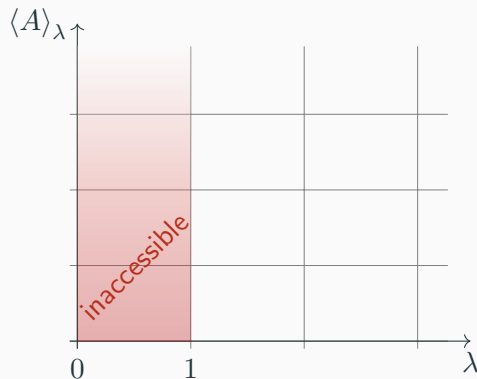


Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$

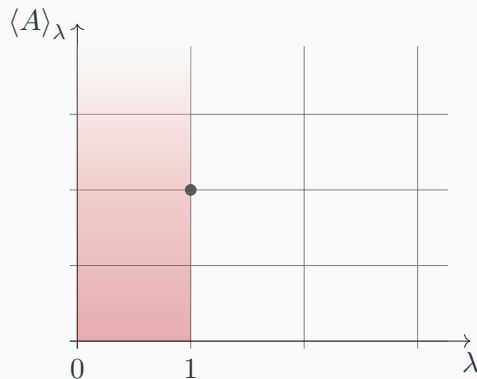


Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$

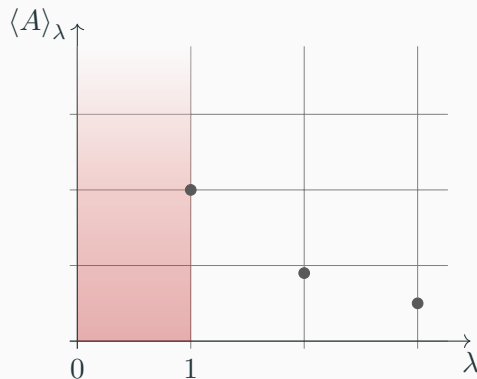


Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$

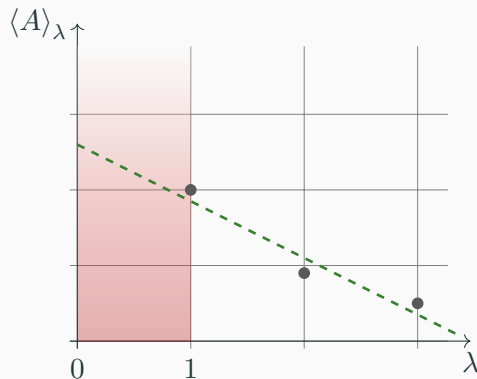


Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$

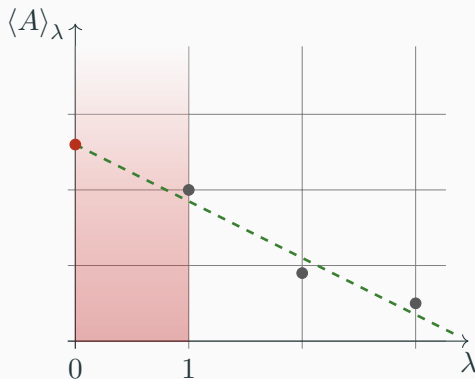


Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$



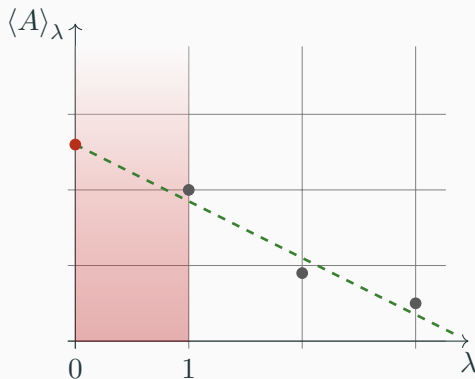
Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

How do we scale the noise **up**?

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$



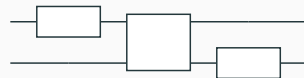
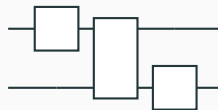
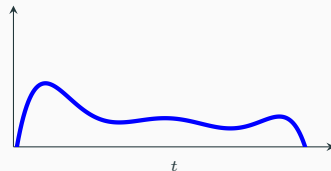
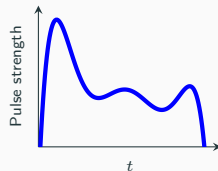
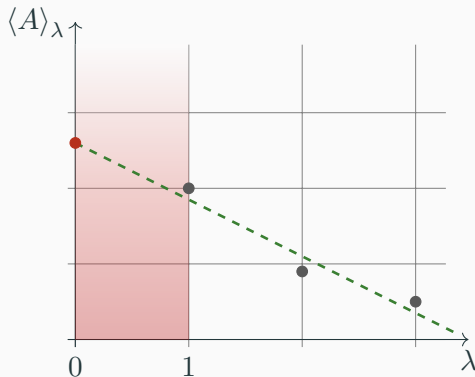
Zero-Noise Extrapolation (ZNE)

Key Idea

Scale noise up, extrapolate back to zero-noise value.

How do we scale the noise **up**?

$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$



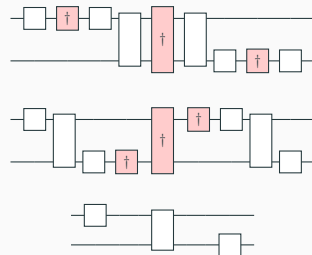
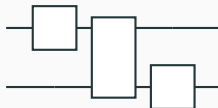
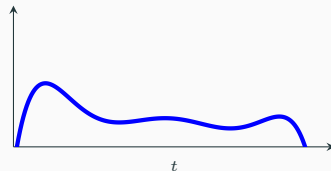
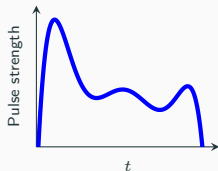
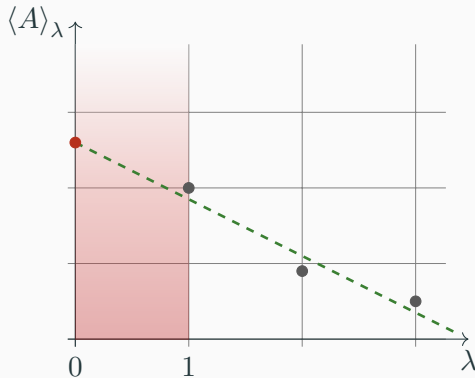
Zero-Noise Extrapolation (ZNE)

Key Idea

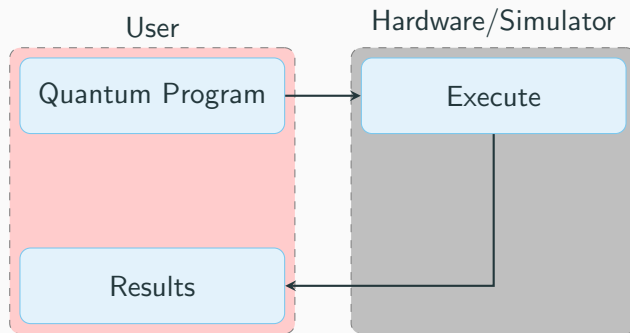
Scale noise up, extrapolate back to zero-noise value.

How do we scale the noise **up**?

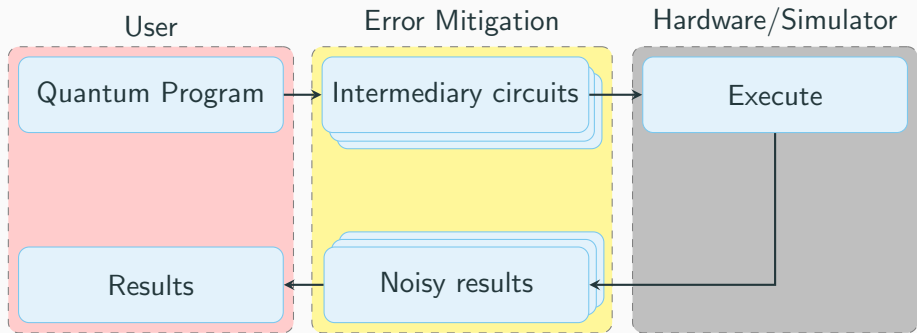
$$\partial_t \rho = -i[H, \rho] + \lambda \mathcal{L}(\rho)$$



Running quantum programs in practice



Running quantum programs in practice with Mitiq



Let's try Mitiq!



<https://github.com/unitaryfund/mitiq-tutorial/>

Executors Continued

An executor is anything with a type signature:

`(QPROGRAM -> QuantumResult)`

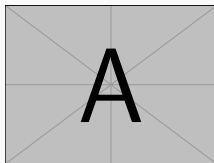


`QuantumResult = float \cup density \cup bitstring`

Sneak Preview of Part II

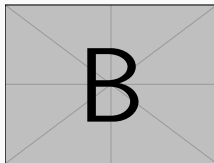
Probabilistic Error Cancellation

Key Idea: Use noisy operations to build up noiseless ones by selective cancellation and sampling.



Digital Dynamical Decoupling

Key Idea: The devil finds work for idle [qubits].



Interested in this work?

OPEN POSITIONS

Community Manager, Full Time, Remote → [MORE INFO](#)

Member of Technical Staff, Full Time, Remote → [MORE INFO](#)

Quantum Open Source Fellow, Full Time, Remote → [MORE INFO](#)



<https://unitary.fund/careers/>