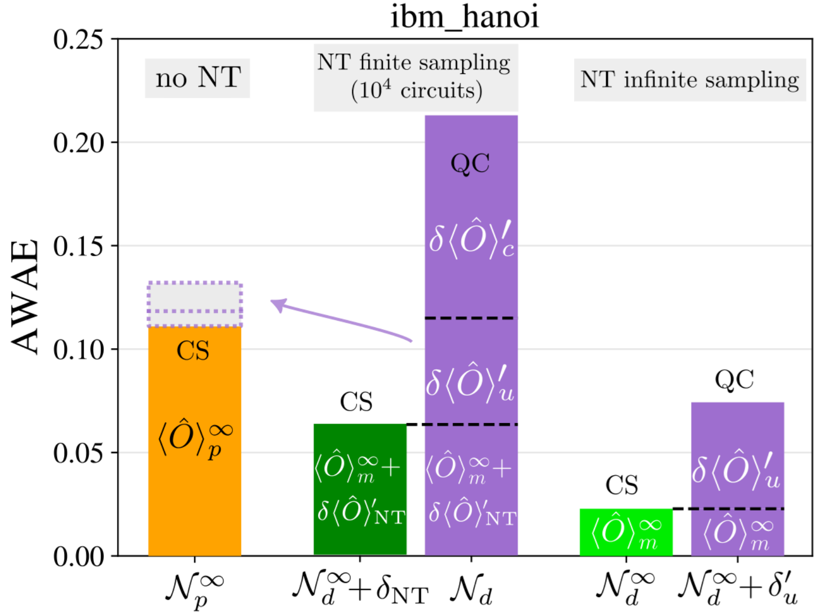
Kyrylo Snizhko
CEA Grenoble

Noise tailoring for error mitigation and for diagnosing quantum computers



TQCI seminar on Quantum Benchmarks
24.06.2025, Paris

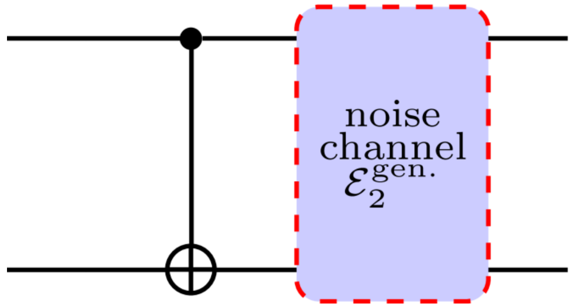
FRANCE HYBRID HPCQUANTUM INITIATIVE
BACQ MetriQs-France
QC4BW
GEFÖRDERT VON Baden-Württemberg MINISTERIUM FÜR WIRTSCHAFT, ARBEIT UND TOURISMUS

Noise in quantum computers is detrimental

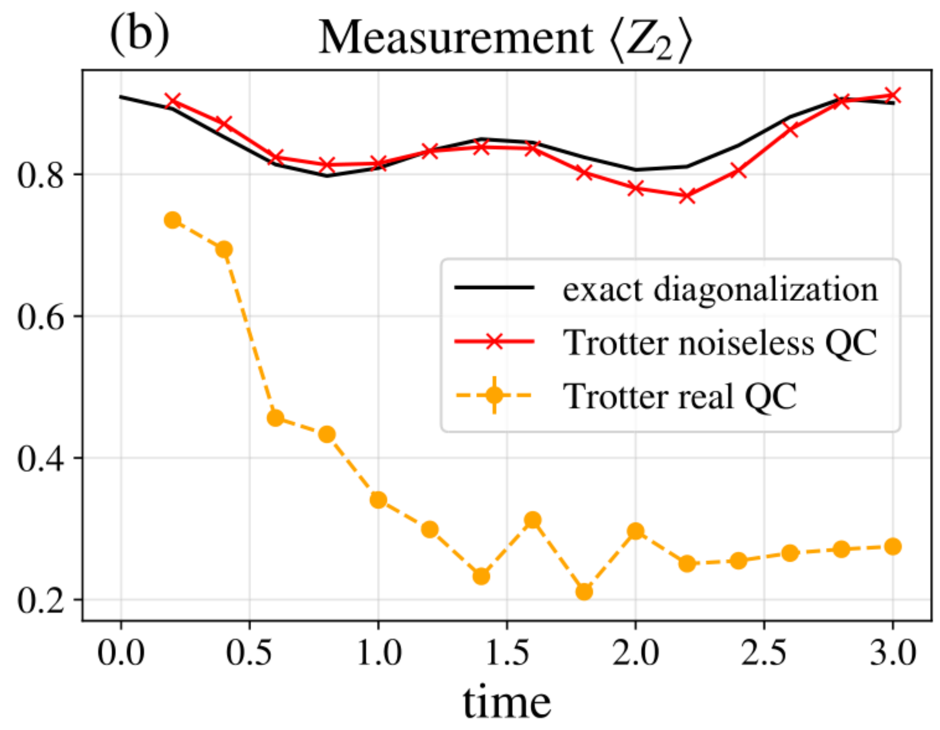
Mitigating crosstalk errors by randomized compiling: Simulation of the BCS model on a superconducting quantum computer

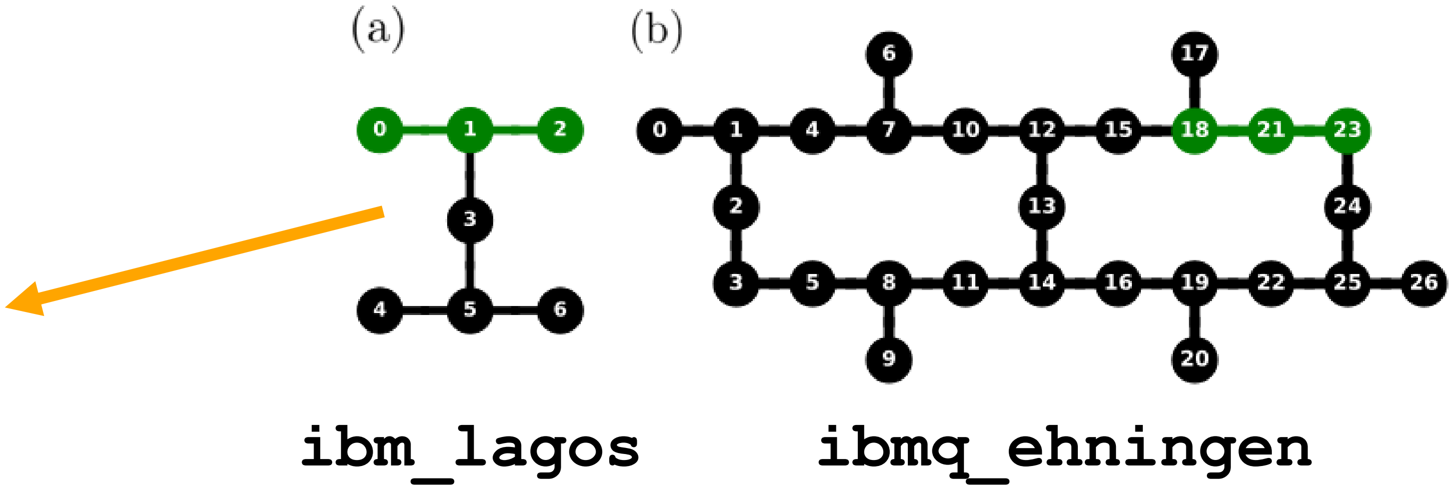
Hugo Perrin[1,*,†] Thibault Scoquart[1,*,‡] Alexander Shnirman,[1,2] Jörg Schmalian,[1,2] and Kyrylo Snizhko[3]
[1]*Karlsruhe Institute of Technology, Institut für Theorie der Kondensierten Materie, TKM, 76049 Karlsruhe, Germany*
[2]*Karlsruher Institut für Technologie, Institut für Quantenmaterialien und Technologien, IQMT, 76021 Karlsruhe, Germany*
[3]*Univ. Grenoble Alpes, CEA, Grenoble INP, IRIG, PHELIQS, 38000 Grenoble, France*

PRResearch **6**, 013142 (2024)



$$\hat{H}_{\text{BCS}} = -\sum_{j=0}^{L-1}(\epsilon_j - \frac{g}{2})\sigma_j^z - \frac{g}{2}\sum_{\substack{i,j=0\\i<j}}^{L-1}\left(\sigma_i^x\sigma_j^x + \sigma_i^y\sigma_j^y\right)$$



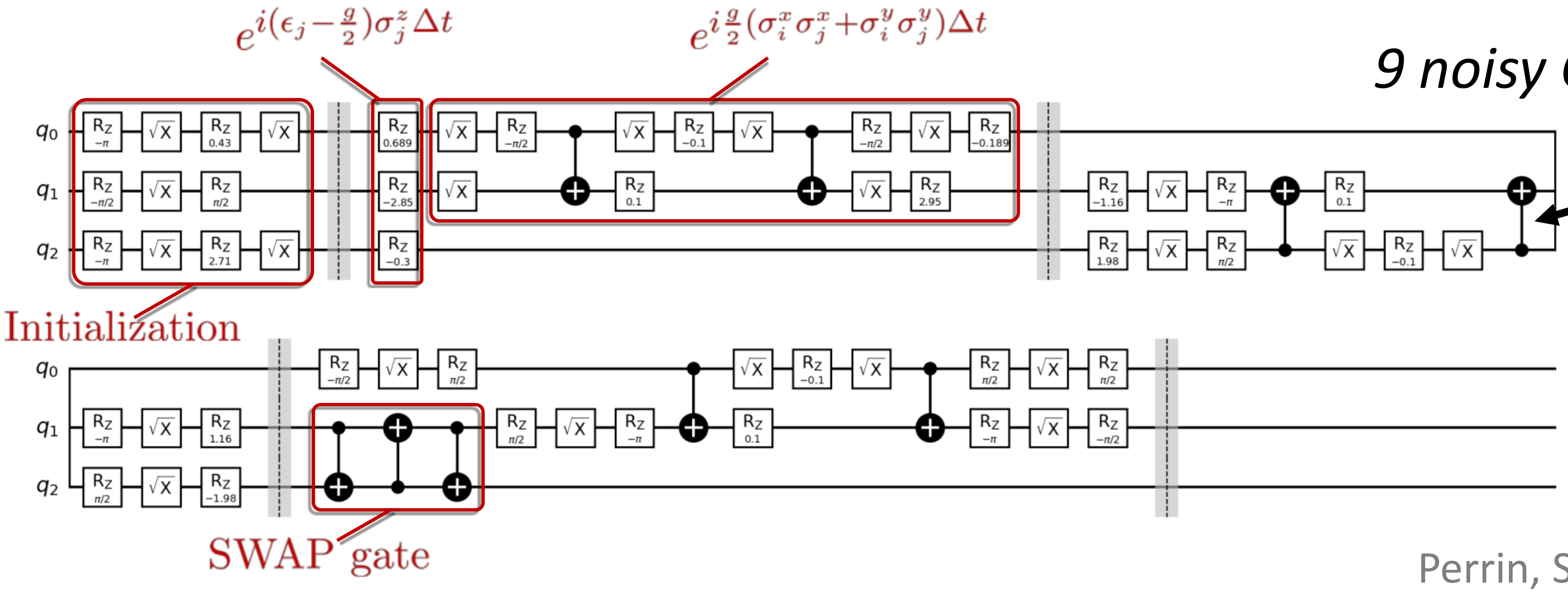


Simulating quench dynamics in the BCS model

$$\hat{H}_{\mathrm{BCS}} = -\sum_{j=0}^{L-1}(\epsilon_j - \frac{g}{2})\sigma_j^z - \frac{g}{2}\sum_{\substack{i,j=0 \\ i<j}}^{L-1}\left(\sigma_i^x\sigma_j^x + \sigma_i^y\sigma_j^y\right)$$

$L = 3$ qubits/Cooper pairs

$$\hat{U}_{\mathrm{BCS}}(\Delta t) \simeq \prod_j e^{i(e_j - \frac{g}{2})\sigma_j^z \Delta t} \prod_{i<j} e^{i\frac{g}{2}(\sigma_i^x\sigma_j^x + \sigma_i^y\sigma_j^y)\Delta t}$$



9 noisy CNOT gates per Trotter step

99% accurate CNOTs

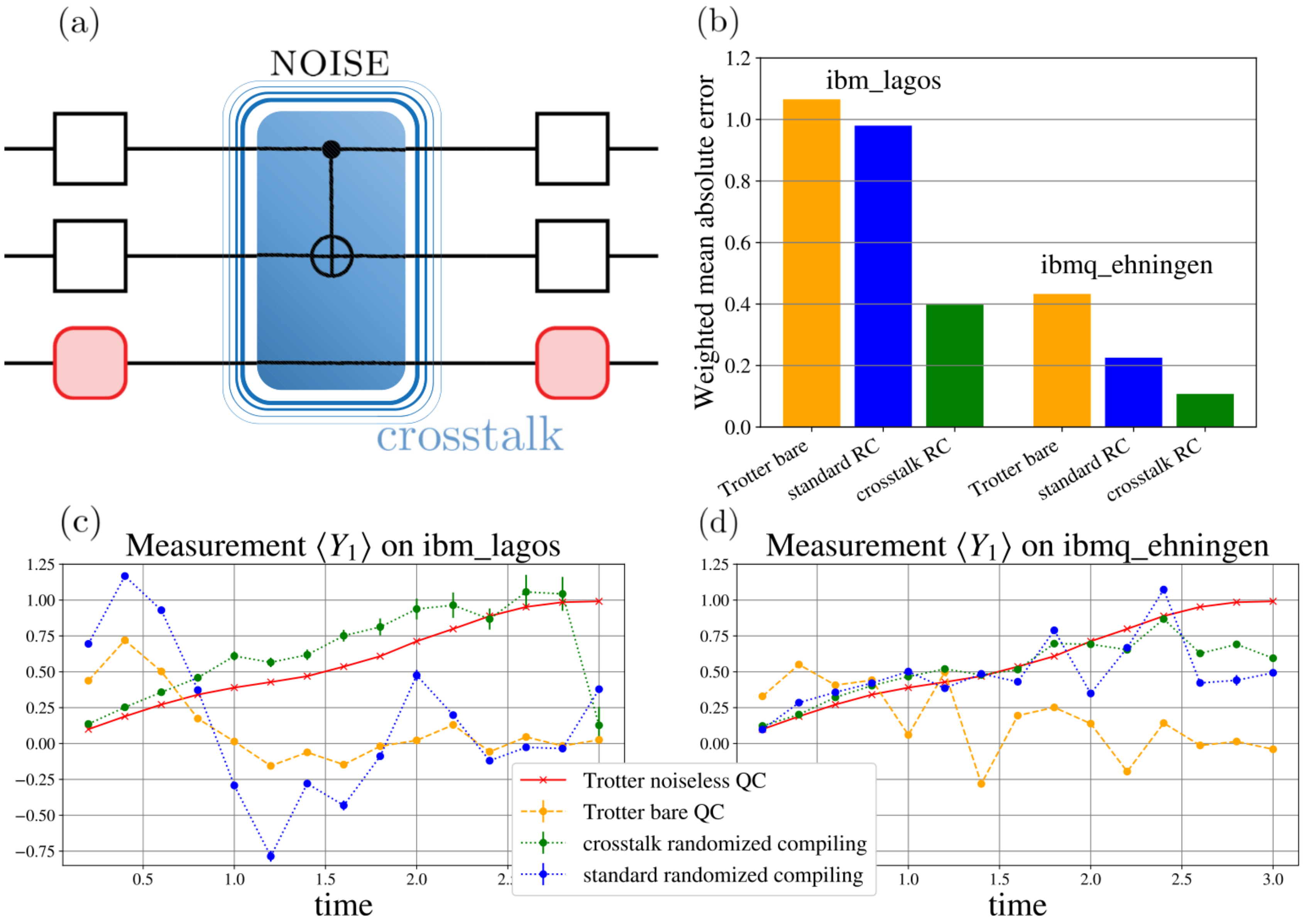
$\Rightarrow$ 1% error per CNOT

$\Rightarrow$ 9% error per Trotter step

$\Rightarrow$ **expect complete noise after $\sim$11 steps**

Perrin, Scoquart, ..., KS PRResearch **6**, 013142 (2024)

Mitigating the noise and seeing the crosstalk



⇐ expect complete noise after ∼11 steps

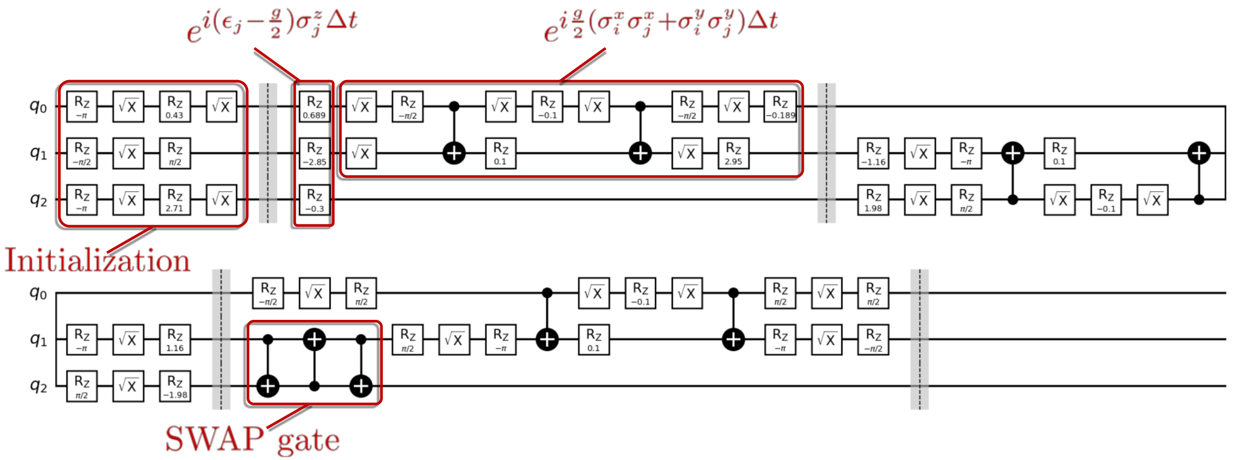
Perrin, Scoquart, …, KS PRResearch **6**, 013142 (2024)

Mitigation: noise estimation circuit (NEC)

Example: Assume N-qubit, **global** depolarizing noise:

$$\mathcal{E}_n^{\text{glob.}}(\rho) = (1-\lambda_{\text{glob.}})\rho + \lambda_{\text{glob.}}\frac{\mathbb{I}}{2^N} \longrightarrow \langle\mathcal{O}\rangle_{\text{noisy}} = (1-\lambda_{\text{glob.}})^{n_{\text{CNOT}}}\langle\mathcal{O}\rangle_{\text{noiseless}}$$

Multiplicative error removed by running **circuits with only the CNOT skeleton:**



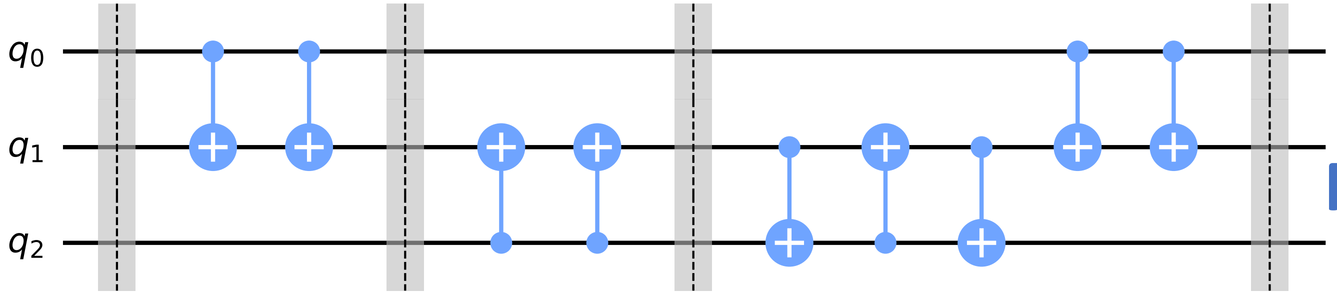
Urbanek et al., PRL **127**, 270502 (2021)

Mitigation: noise estimation circuit (NEC)

Example: Assume N-qubit, **global** depolarizing noise:

$$\mathcal{E}_n^{\text{glob.}}(\rho) = (1-\lambda_{\text{glob.}})\rho + \lambda_{\text{glob.}}\frac{\mathbb{I}}{2^N} \longrightarrow \langle\mathcal{O}\rangle_{\text{noisy}} = (1-\lambda_{\text{glob.}})^{n_{\text{CNOT}}}\langle\mathcal{O}\rangle_{\text{noiseless}}$$

Multiplicative error removed by running **circuits with only the CNOT skeleton:**



Noise estimation circuit

=

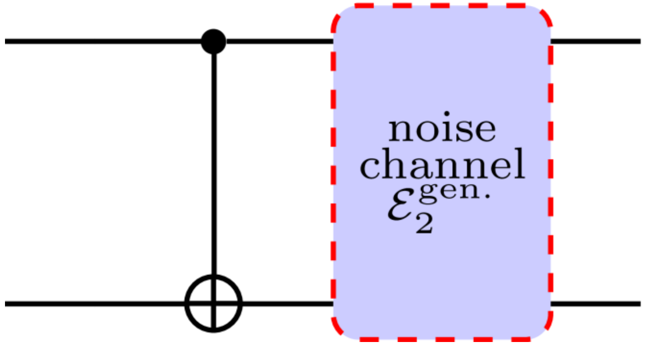
Trotter step circuit without single qubit gates

$$\langle\hat{O}\rangle_{\text{mitigated}} = \frac{\langle\hat{O}\rangle_{\text{measured}}}{\mathcal{F}_{\text{NEC}}} \sim \frac{\langle\hat{O}\rangle_{\text{measured}}}{\langle 1\rangle_{\text{measured}}}$$

Urbanek et al., PRL **127**, 270502 (2021)

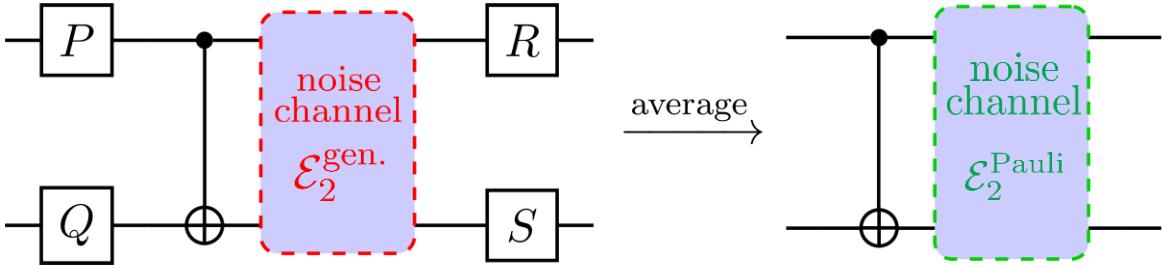
Simplifying the noise structure: randomized compiling (RC)

- General noise:



$$\mathcal{E}_2^{\text{gen.}}(\rho) = \sum_{i,j,k,l=0}^{3} p_{ijkl}(\sigma^i \otimes \sigma^j).\rho.(\sigma^k \otimes \sigma^l)$$

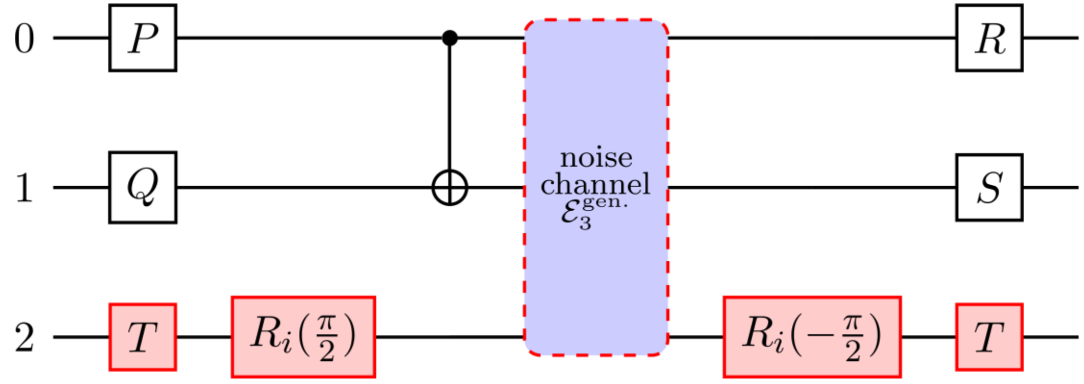
- Randomized compiling to get Pauli noise



$$\mathcal{E}_2^{\text{pauli}}(\rho) = \sum_{i,j=0}^{3} p_{ij}(\sigma^i \otimes \sigma^j).\rho.(\sigma^i \otimes \sigma^j)$$

Kern et al., Eur. Phys. J. D **32**, 153 (2005)
Wallman, Emerson, PRA **94**, 052325 (2016)
Hashim et al., PRX **11**, 041039 (2021)

Crosstalk!

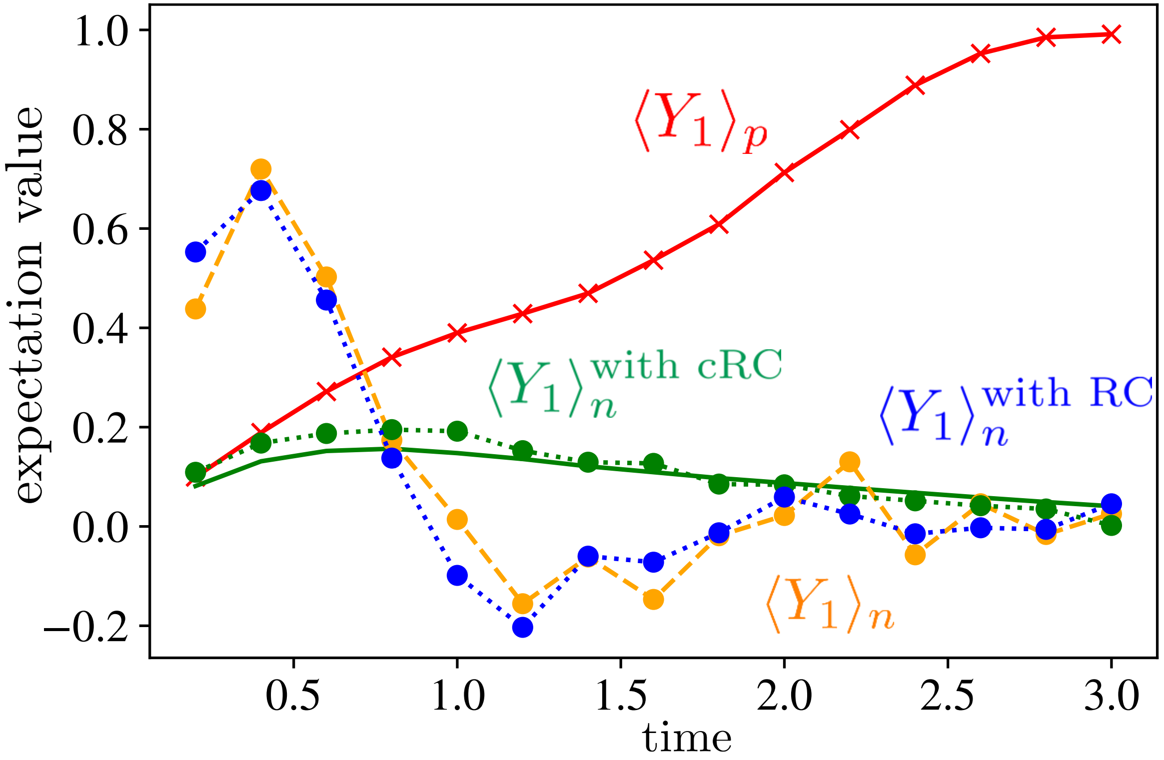


Crosstalk RC (cRC) average

$$\mathcal{E}_1^{\text{dep.}}(\rho) = \left(1-\lambda'\right)\rho + \lambda' \frac{\mathbb{I}_2}{2}$$

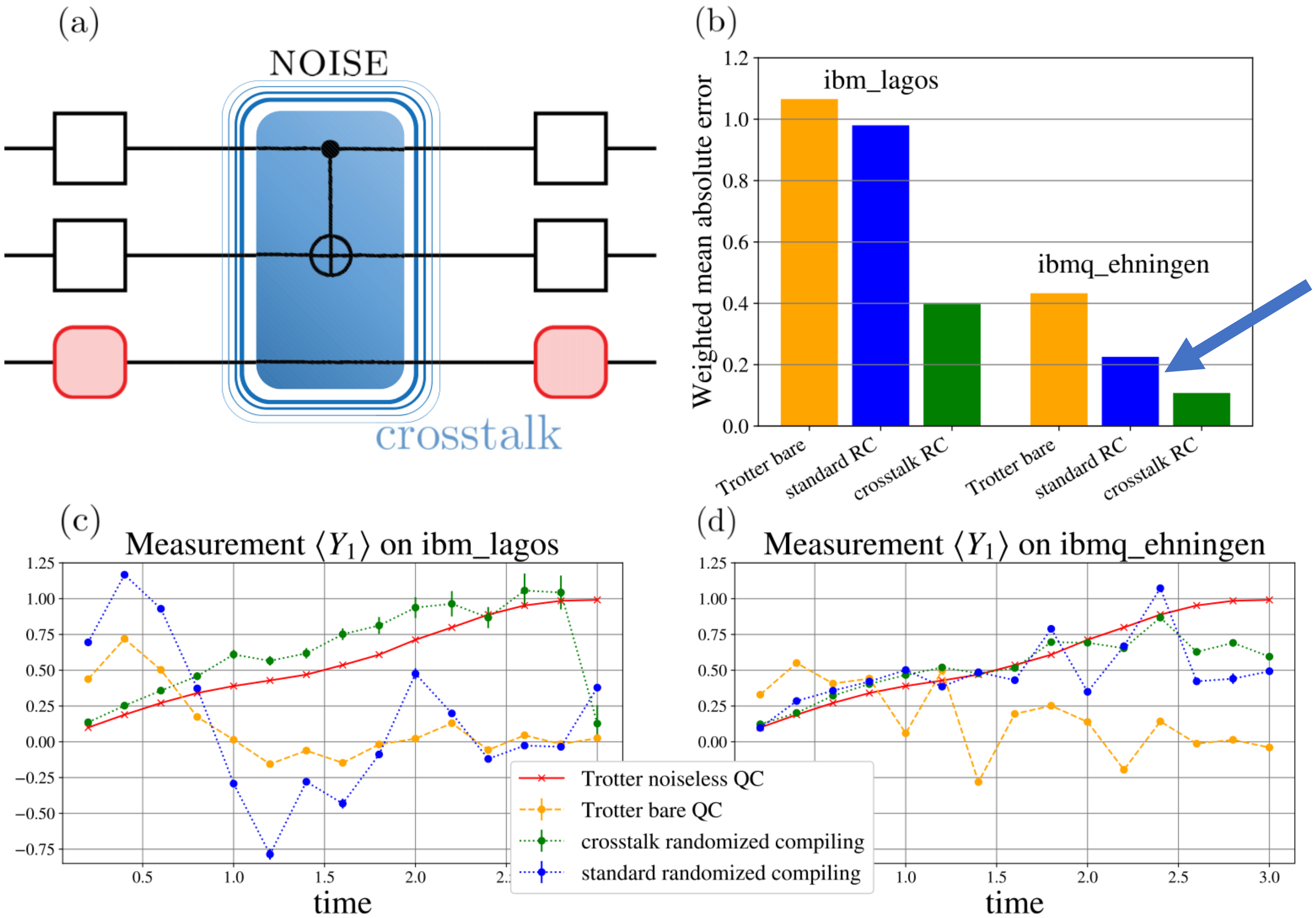
Yields **depolarizing noise** on the **neighbouring qubit !**

RC (no NEC):



Perrin, Scoquart, …, KS PRResearch **6**, 013142 (2024)

RC + NEC = improved results + error characterization



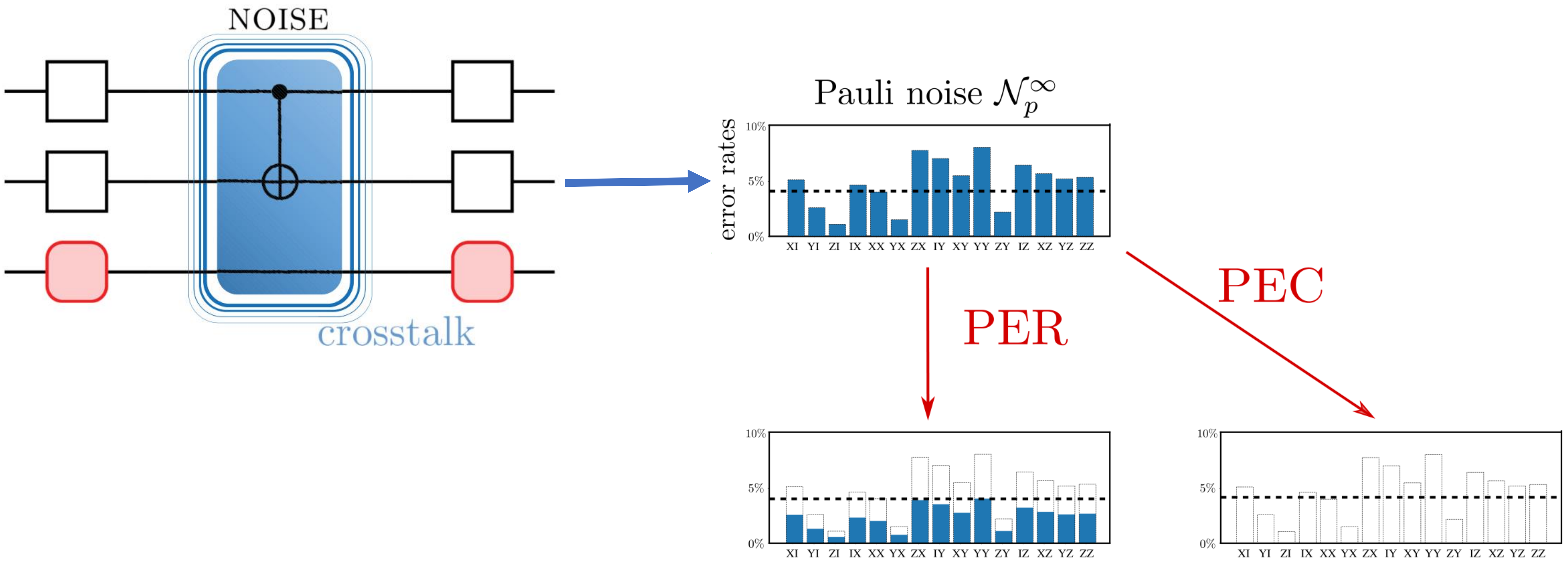
Stack of techniques

Direct estimation of spillover crosstalk by RC vs cRC

⇐ **NEC used for all curves**

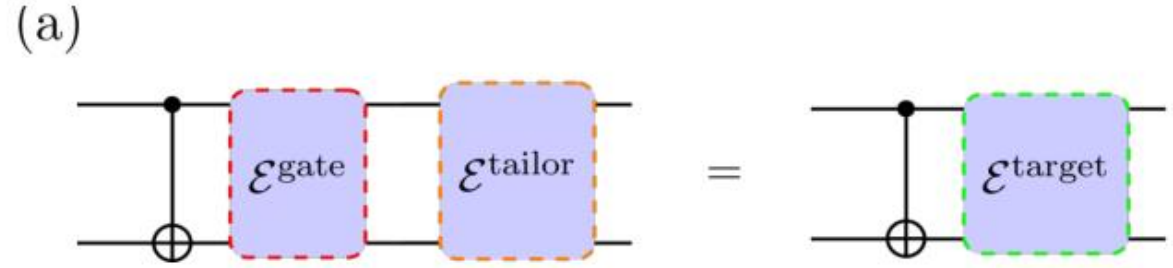
Perrin, Scoquart, …, KS PRResearch **6**, 013142 (2024)

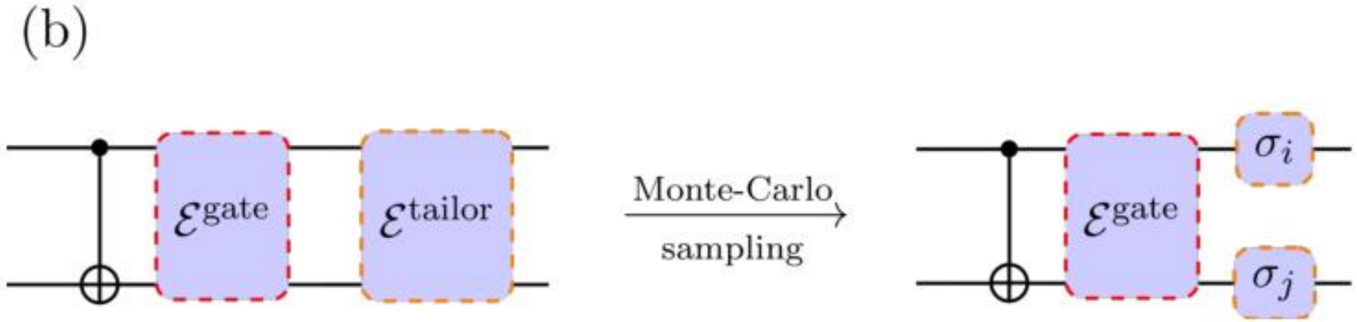
Probabilistic error cancellation (PEC)
reduction (PER)



Temme, Bravyi, Gambetta, PRL **119**, 180509 (2017)
Berg, Minev, Kandala, Temme, Nature Physics **19**, 1116 (2023)

How do PEC and PER work?



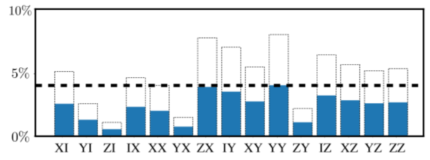


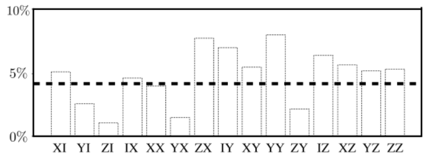
Sample $(\sigma_i, \sigma_j) = P_a$

Get target noise channel

PER

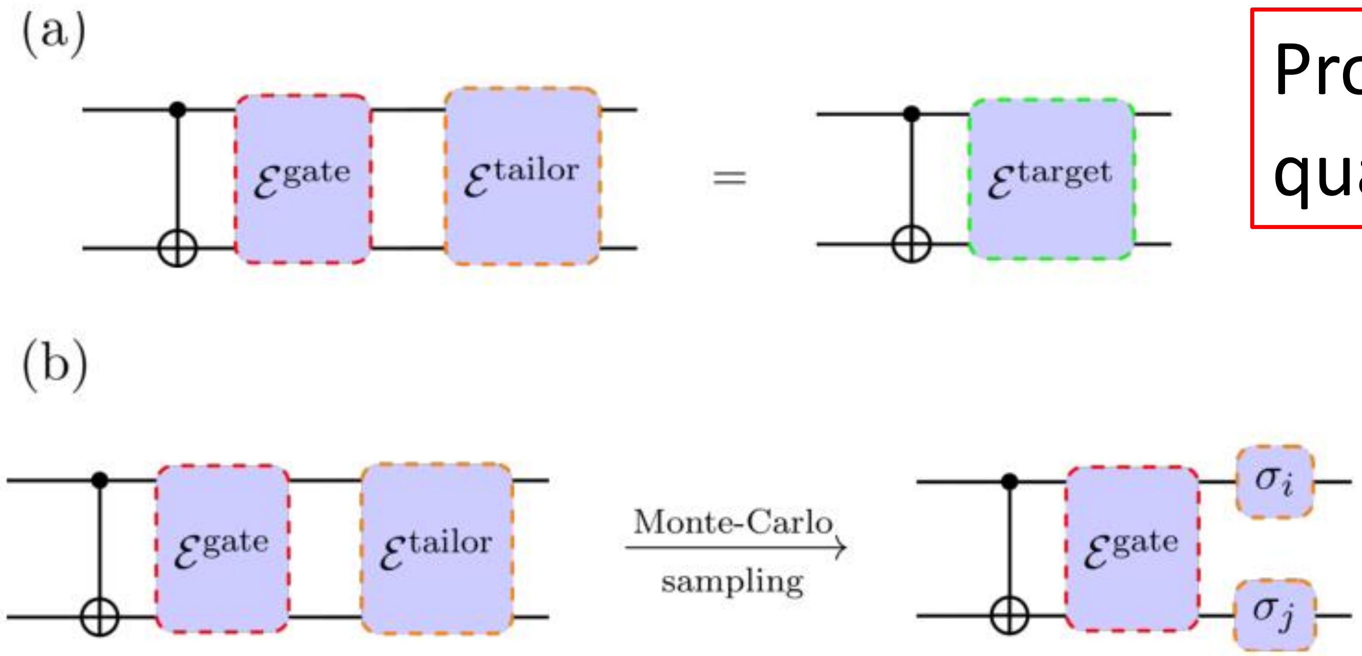
PEC





Temme, Bravyi, Gambetta, PRL **119**, 180509 (2017)
Berg, Minev, Kandala, Temme, Nature Physics **19**, 1116 (2023)

How do PEC and PER work?



Sample $(\sigma_i, \sigma_j) = P_a$

Problem: need to sample quasiprobability distribution q_a^{tailor}

Sample:

$$\tilde{p}_a^{\text{tailor}} = \frac{|q_a^{\text{tailor}}|}{\gamma},$$

$$\gamma = \sum_a |q_a^{\text{tailor}}| > 1$$

Take the sign q_a^{tailor} into account when calculating the Monte-Carlo average

Multiply the average by $\gamma^{n_{\text{CNOT}}}$

⇒ exponentially costly to keep a fixed error bar (sign problem)

Temme, Bravyi, Gambetta, PRL **119**, 180509 (2017)

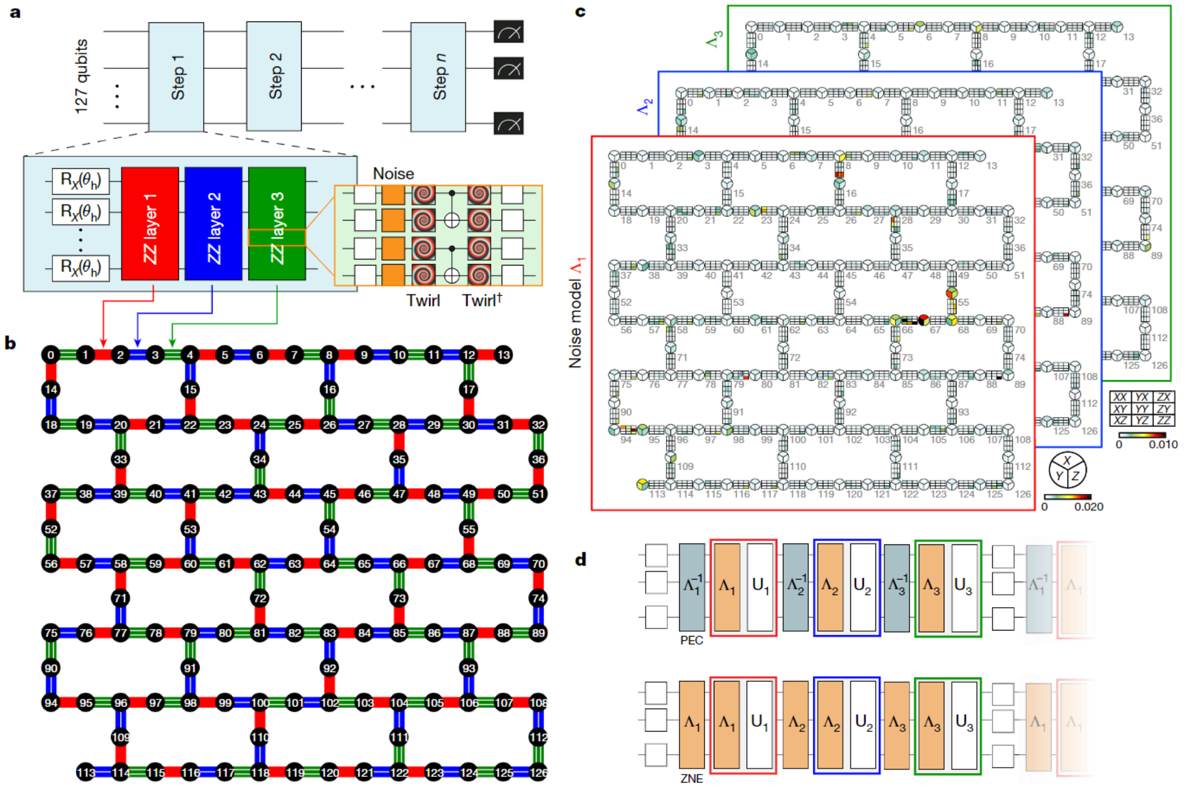
Berg, Minev, Kandala, Temme, Nature Physics **19**, 1116 (2023)

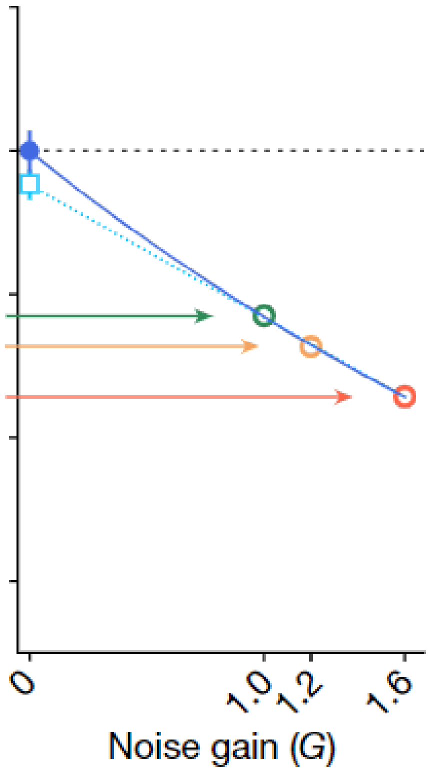
How do PEC and PER work?

$\boxed{\text{Multiply the average by } \gamma^{n_{\mathrm{CNOT}}}}$ $\Rightarrow$ **exponentially costly when reducing noise**

Possible solution: increase noise (avoid quasiprobability distributions!)

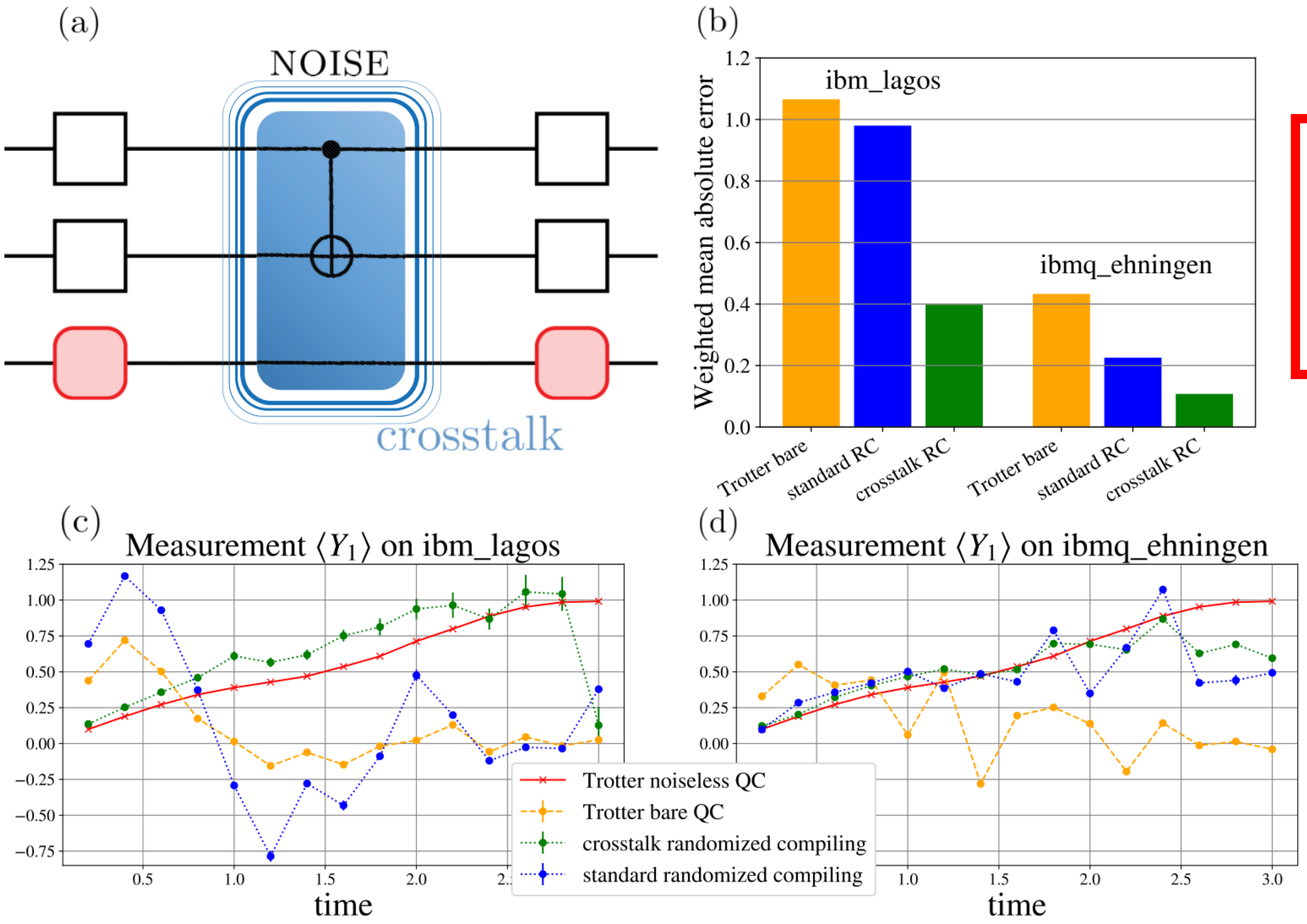
E.g., IBM “quantum utility” experiment:





Kim et al., Nature **618**, 500 (2023)

RC + NEC = improved results + error characterization



NEC requires global depolarizing noise

$$\mathcal{E}_n^{\text{glob.}}(\rho) = (1-\lambda_{\text{glob.}})\rho + \lambda_{\text{glob.}}\frac{\mathbb{I}}{2^N}$$

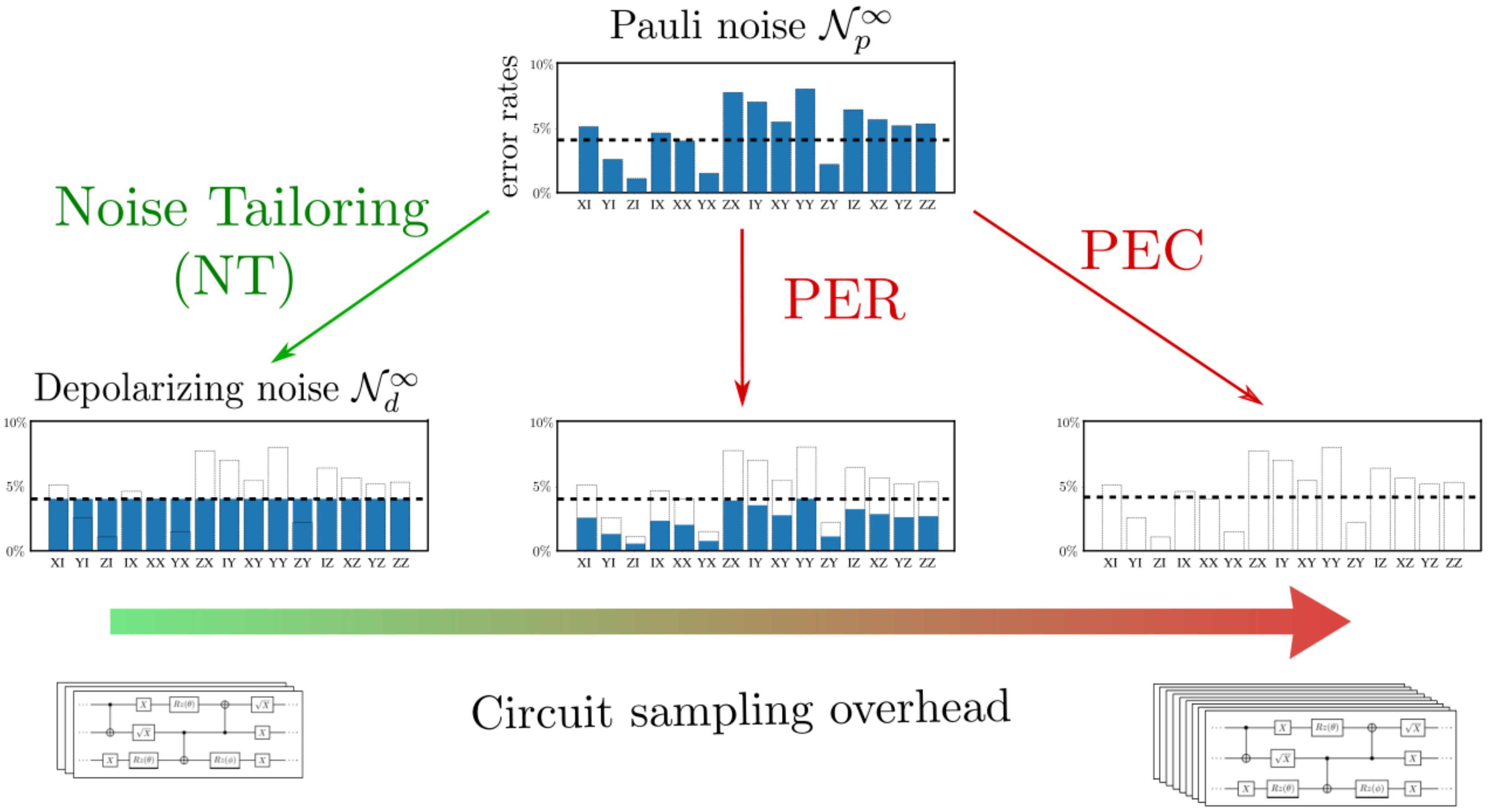
⇐ NEC used for all curves

Perrin, Scoquart, ..., KS PRResearch **6**, 013142 (2024)

Why not change the noise shape?

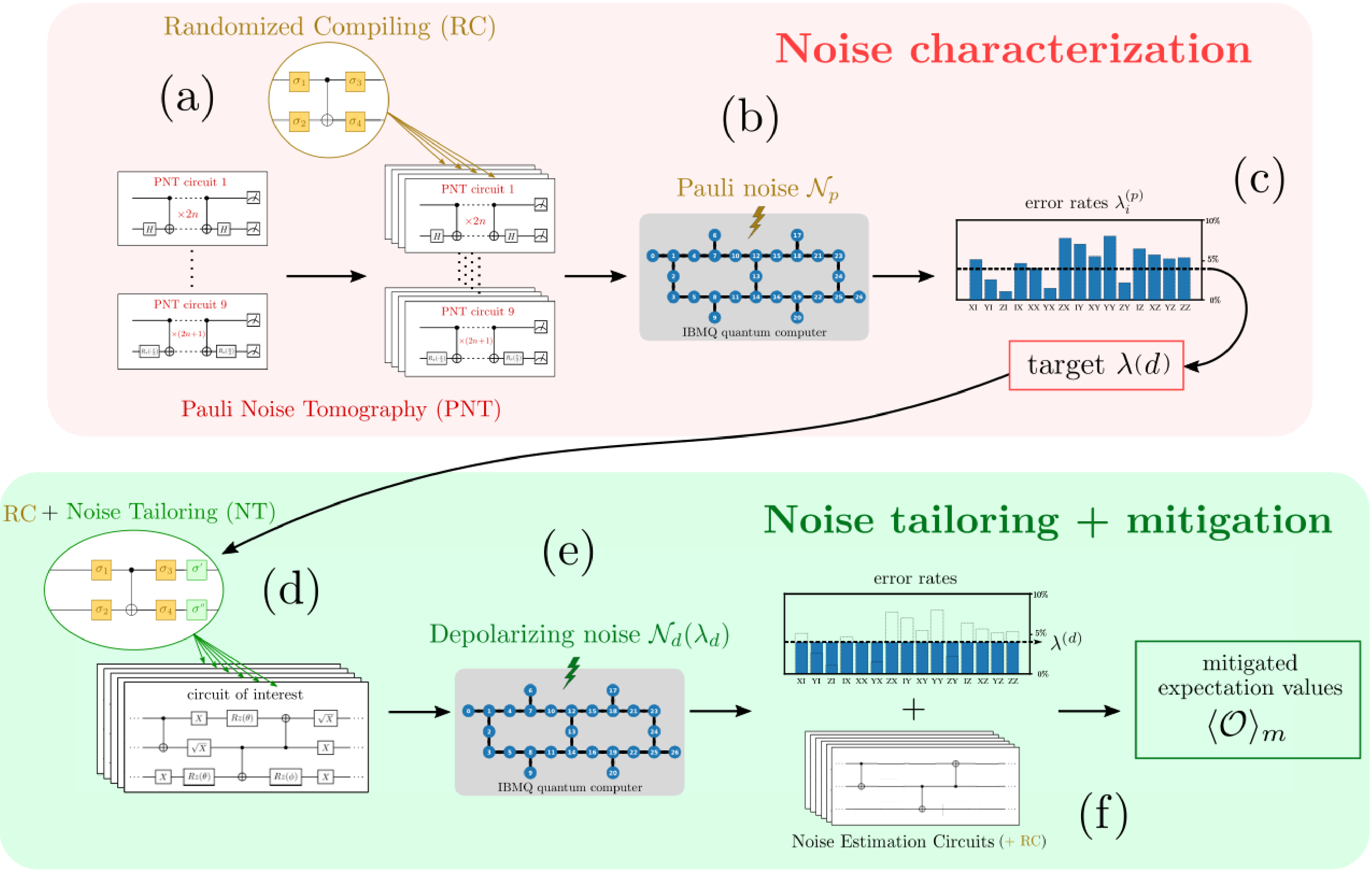
NEC requires global **depolarizing** noise

$$\mathcal{E}_n^{\text{glob.}}(\rho) = (1-\lambda_{\text{glob.}})\rho + \lambda_{\text{glob.}}\frac{\mathbb{I}}{2^N}$$



Perrin, Scoquart, KS unpublished

Noise tailoring (NT) protocol



Perrin, Scoquart, KS unpublished

Choosing the target noise

Our target: depolarizing noise.

$$\mathcal{N}_d^{\infty}(\rho) = (1 - 15\lambda_d)\rho + \lambda_d \sum_{i=1}^{15} \sigma_i^{2-\mathrm{qubit}} \rho \sigma_i^{2-\mathrm{qubit}}$$

Reason: NEC mitigation

$$\langle \hat{O} \rangle_{\mathrm{mitigated}} = \frac{\langle \hat{O} \rangle_{\mathrm{measured}}}{\mathcal{F}_{\mathrm{NEC}}} \sim \frac{\langle \hat{O} \rangle_{\mathrm{measured}}}{\langle 1 \rangle_{\mathrm{measured}}}$$

The effect of finite circuit («trajectory») sampling:

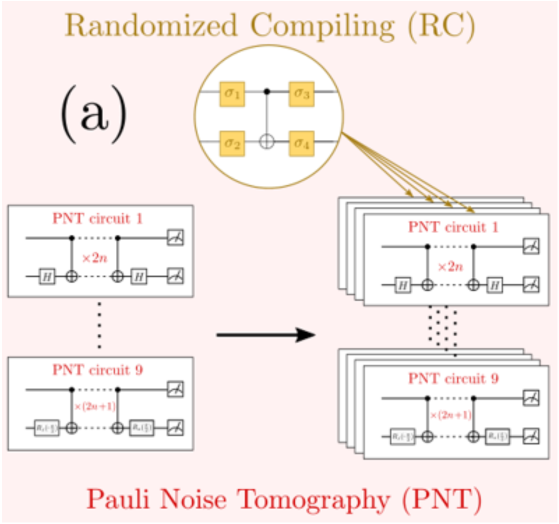
$$\langle \hat{O} \rangle_{\mathrm{mitigated}} = \langle \hat{O} \rangle_{\mathrm{mitigated}}^{\infty} + \frac{\gamma^{n_{\mathrm{CNOT}}}}{\mathcal{F}_{\mathrm{NEC}}} [\text{Sampling errors}]$$

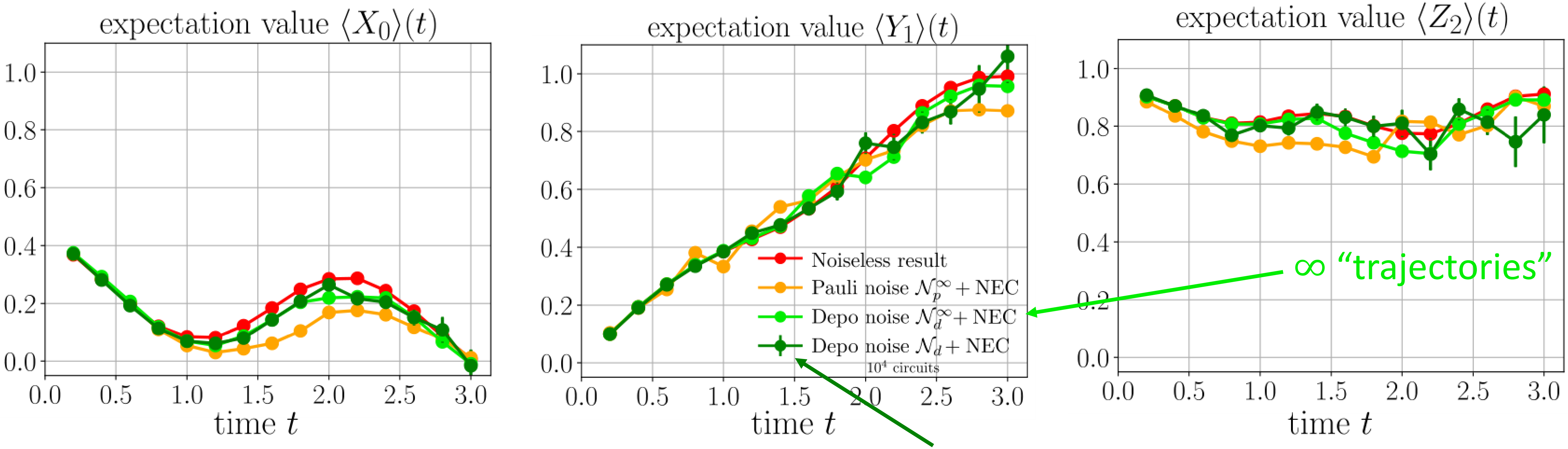
Choose λ_d to minimize $\frac{\gamma^{n_{\mathrm{CNOT}}}}{\mathcal{F}_{NEC}}$

Perrin, Scoquart, KS unpublished

Classical simulations of NT

Noise extracted from actual
IBM quantum computer (**`ibm_hanoi`**, 27 qubits)





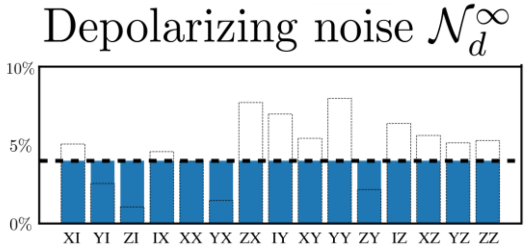
∞ "trajectories"

10000 "trajectories"

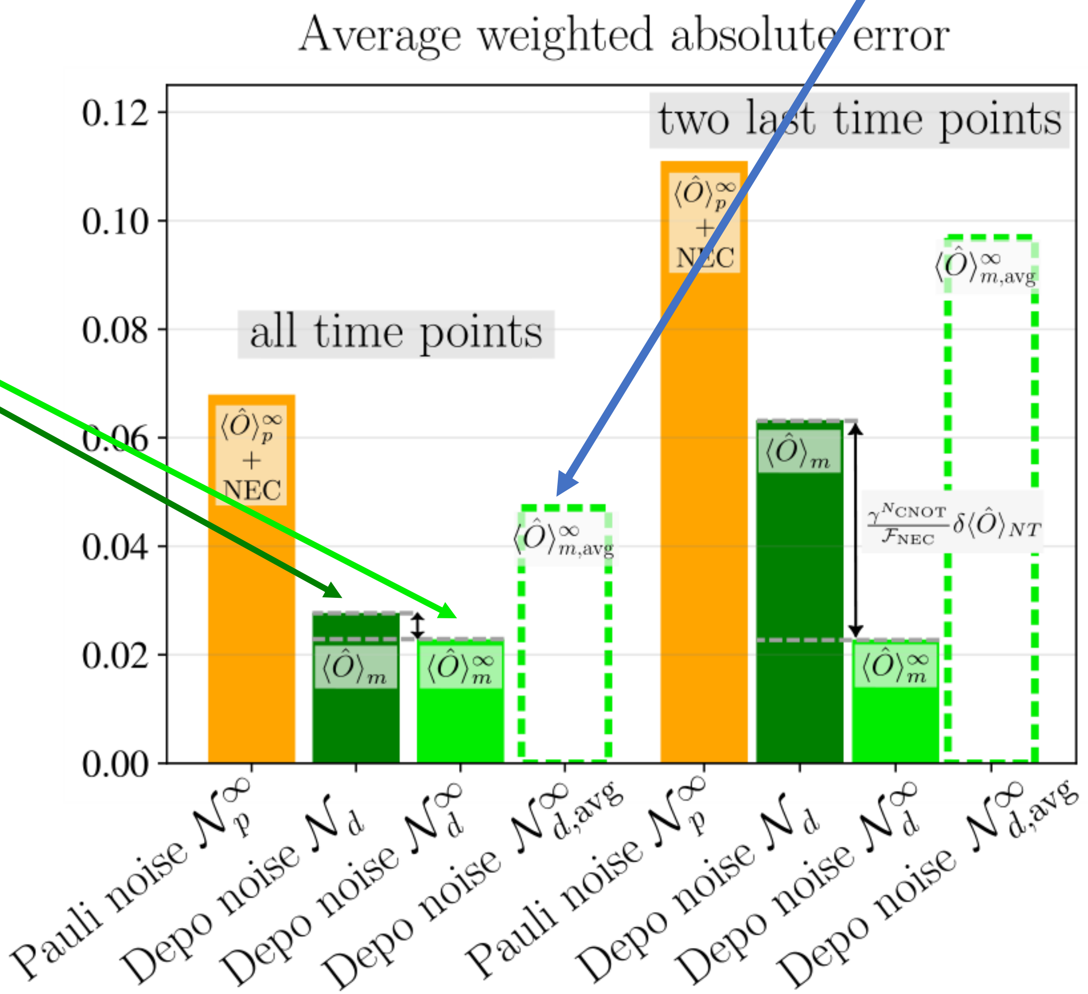
Perrin, Scoquart, KS unpublished

Classical simulations of NT

Choose average λ_d

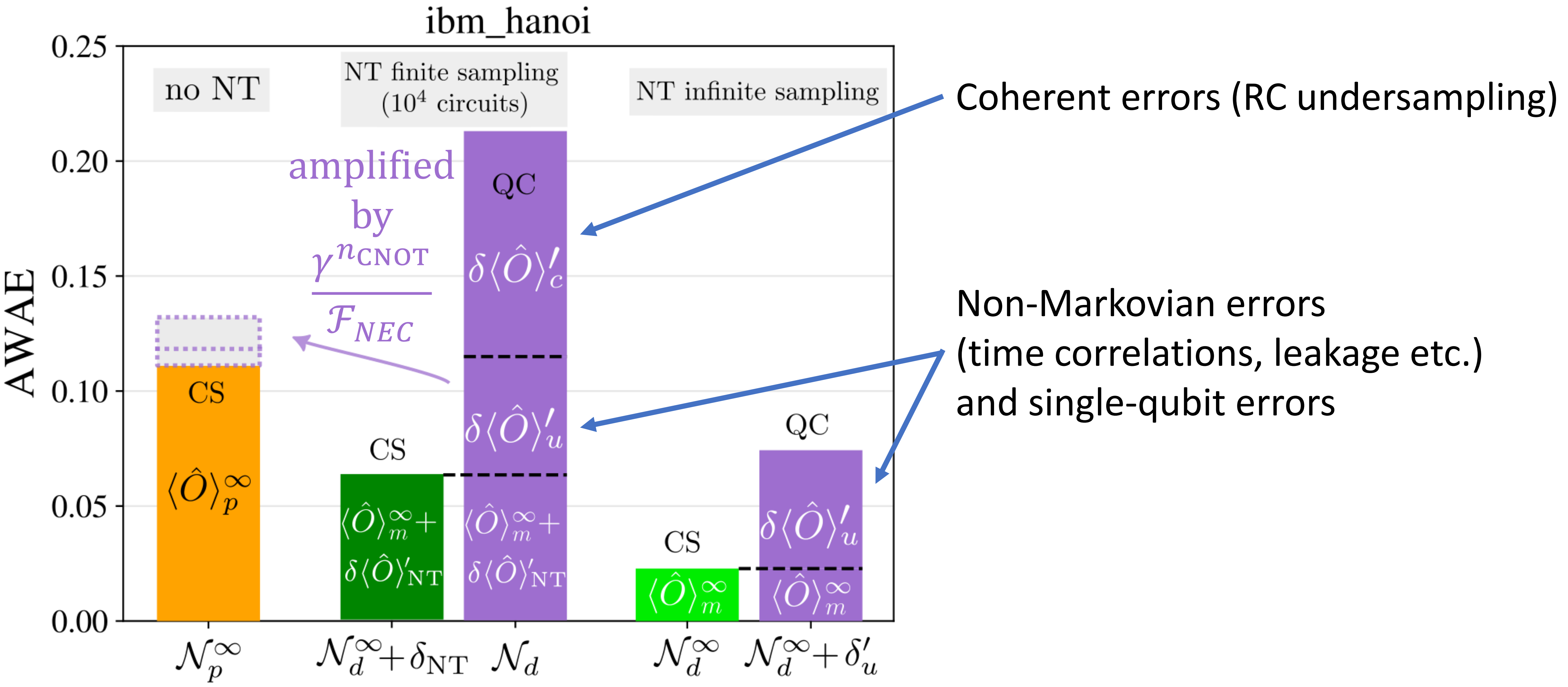


Choose λ_d to minimize $\dfrac{\gamma^{n_{\text{CNOT}}}}{\mathcal{F}_{NEC}}$



Perrin, Scoquart, KS unpublished

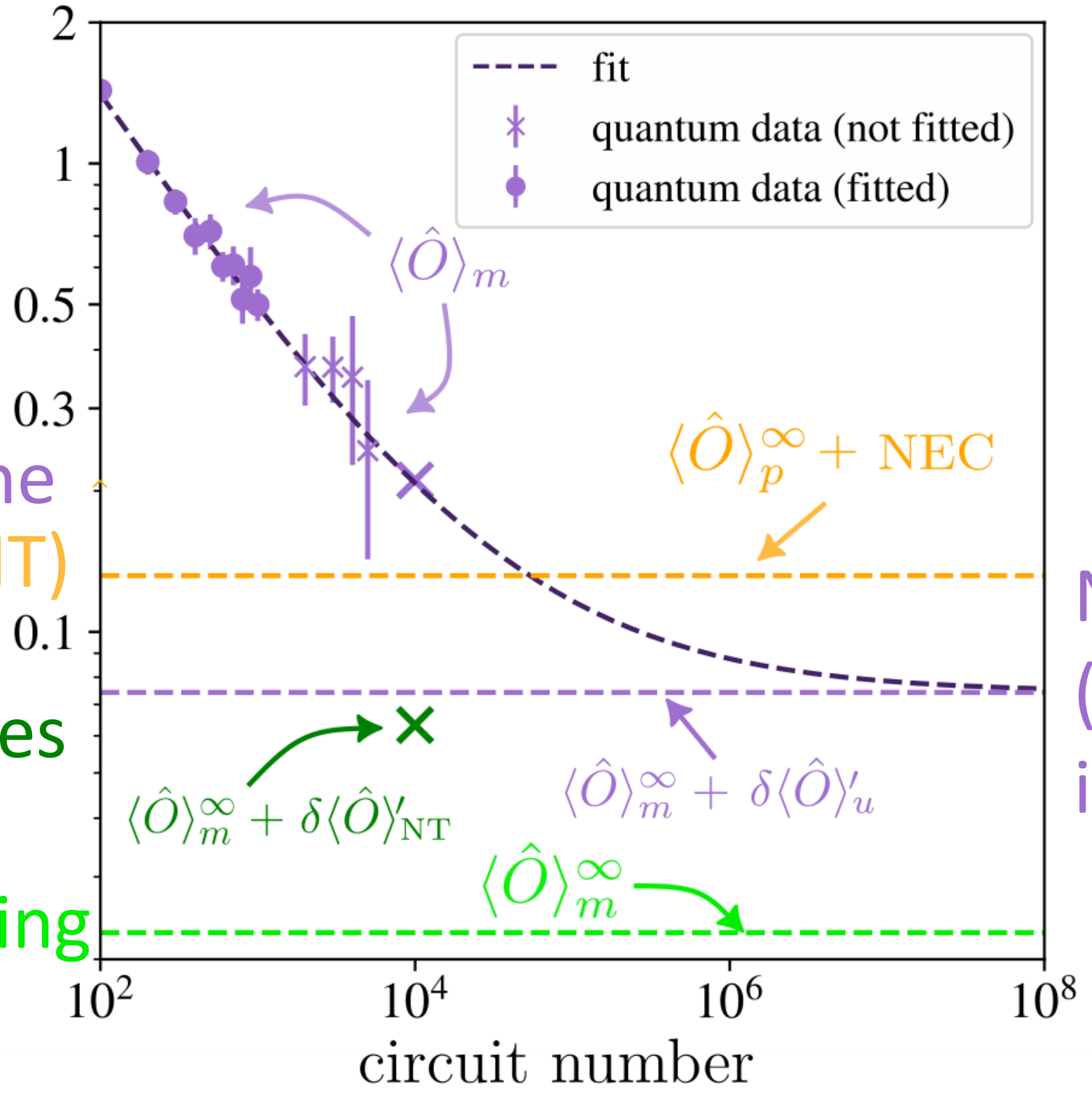
NT on an actual quantum computer



Perrin, Scoquart, KS unpublished

How to estimate accuracy at infinite sampling on a quantum computer?

Bootstrapping



NT on a real machine

RC+NEC (no NT)

NT: CS 10^4 trajectories

NT: CS infinite sampling

NT on a real machine (extrapolation to infinite sampling)

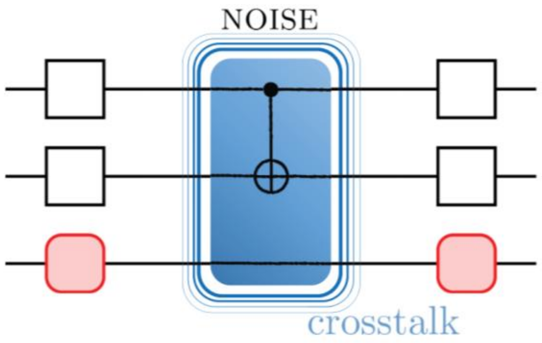
Perrin, Scoquart, KS unpublished

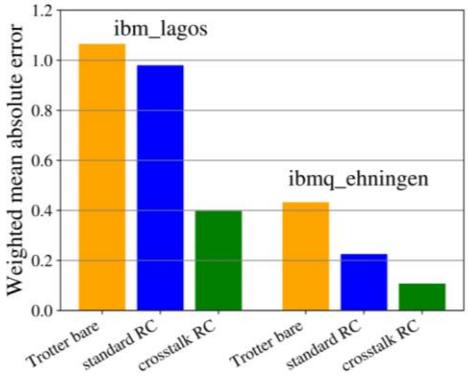
Conclusions

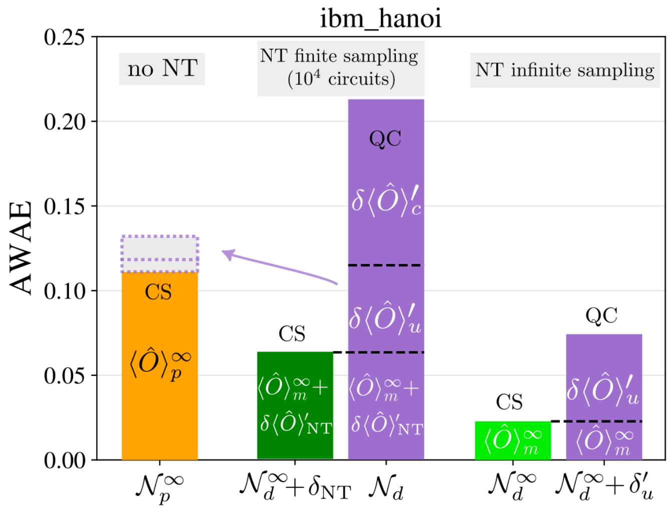
BACQ
MetriQs-France

Application-based benchmarking
gives insight about error sources

RC vs cRC to
characterise crosstalk



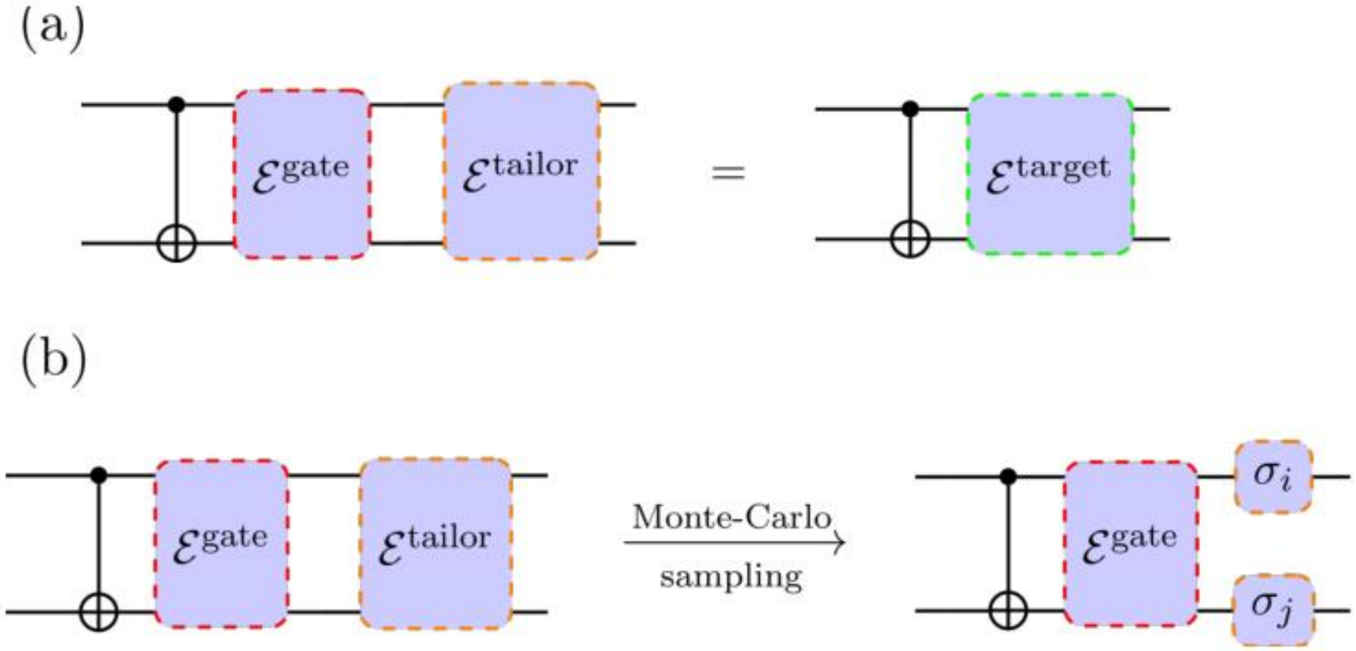




Noise tailoring to
characterize (small) but important sources of noise
(residual coherent; single-qubit & non-Markovian)

Thank you for attention!

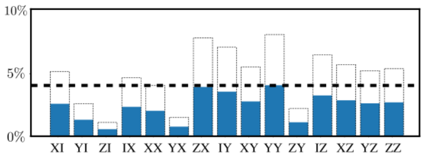
How do PEC and PER work?

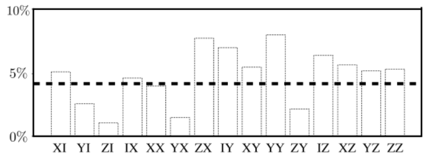


Sample σ_i, σ_j → Get target noise channel

PER

PEC





Temme, Bravyi, Gambetta, PRL **119**, 180509 (2017)
Berg, Minev, Kandala, Temme, Nature Physics **19**, 1116 (2023)

Noise channel representations

χ-matrix representation:

$$\rho \to \mathcal{E}(\rho) = \sum_{a,b} \chi_{ab} P_a \rho P_b$$

with n-qubit Pauli matrices $P_a \in \{I, X, Y, Z\}^{\otimes n_{\text{qubits}}}$

Pauli noise: $\chi_{ab} = p_a \delta_{ab}, \sum_a p_a = 1$

Pauli transfer-matrix (PTM) representation:

$$\rho = \rho_a P_a \qquad (\text{or } \rho_a = \frac{1}{2^{n_{\text{qubits}}}} \mathrm{Tr}(P_a \rho))$$

$$\{\rho_a\} \to \left\{ \sum_b E_{ab} \rho_b \right\}$$

PTM: $E_{ab} = \frac{1}{2^{n_{\text{qubits}}}} \mathrm{Tr}\big(P_a \mathcal{E}(P_b)\big)$

Pauli noise: $E_{ab} = f_a \delta_{ab}$

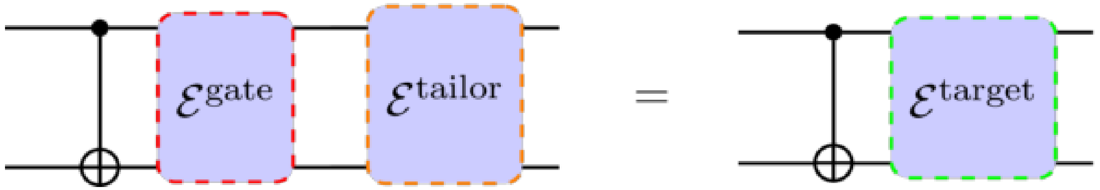
Connection (Walsh-Hadamard transform):

$$p_a = \frac{1}{4^{n_{\text{qubits}}}} \sum_b (-1)^{\langle P_a, P_b \rangle_{\text{sp}}} f_b$$

Symplectic inner product:

$$\langle P_a, P_b \rangle_{\text{sp}} = \begin{cases} 0, & \text{if } P_a P_b = +P_b P_a \\ 1, & \text{if } P_a P_b = -P_b P_a \end{cases}$$

How do PEC and PER work?

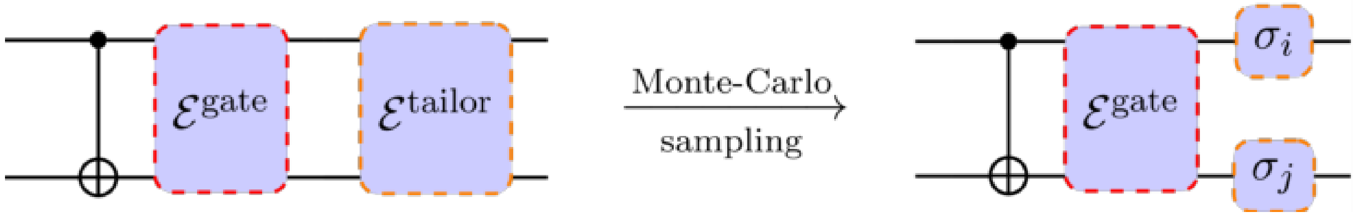


$$E^{\text{target}} = E^{\text{tailor}} \times E^{\text{gate}}$$

$$\boxed{\Rightarrow E^{\text{tailor}} = E^{\text{target}} \times \left(E^{\text{gate}}\right)^{-1} = f_a^{\text{tailor}} \delta_{ab}}$$

$$p_a^{\text{tailor}} = \frac{1}{4^{n_{\text{qubits}}}} \sum_b (-1)^{\langle P_a, P_b \rangle_{\text{sp}}} f_b^{\text{tailor}}$$

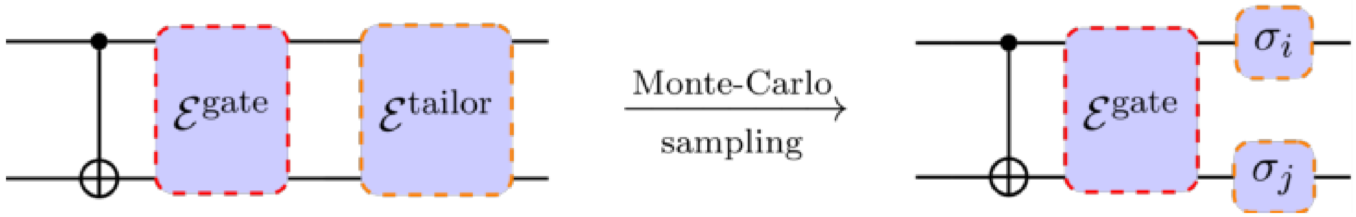
$$\mathcal{E}(\rho) = \sum_a p_a^{\text{tailor}} P_a \rho P_a$$



Temme, Bravyi, Gambetta, PRL **119**, 180509 (2017)
Berg, Minev, Kandala, Temme, Nature Physics **19**, 1116 (2023)

How do PEC and PER work?

$$p_a^{\text{tailor}} = \frac{1}{4^{n_{\text{qubits}}}} \sum_b (-1)^{\langle P_a, P_b \rangle_{\text{sp}}} f_b^{\text{tailor}} \qquad \mathcal{E}(\rho) = \sum_a p_a^{\text{tailor}} P_a \rho P_a$$



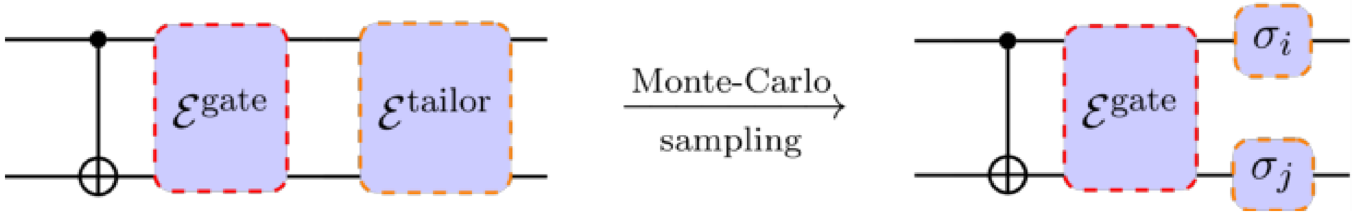
Problem: p_a^{tailor} can be negative

Temme, Bravyi, Gambetta, PRL **119**, 180509 (2017)
Berg, Minev, Kandala, Temme, Nature Physics **19**, 1116 (2023)

How do PEC and PER work?

$$q_a^{\text{tailor}} = \frac{1}{4^{n_{\text{qubits}}}} \sum_b (-1)^{\langle P_a, P_b \rangle_{\text{sp}}} f_b^{\text{tailor}} \qquad \mathcal{E}(\rho) = \sum_a q_a^{\text{tailor}} P_a \rho P_a$$

Sample: $\tilde{p}_a^{\text{tailor}} = \frac{|q_a^{\text{tailor}}|}{\gamma}, \qquad \gamma = \sum_a |q_a^{\text{tailor}}| > 1$



Take the sign q_a^{tailor} into account when calculating the Monte-Carlo average

Multiply the average by $\gamma^{n_{\text{CNOT}}}$

⇒ exponentially costly to keep a fixed error bar (sign problem)

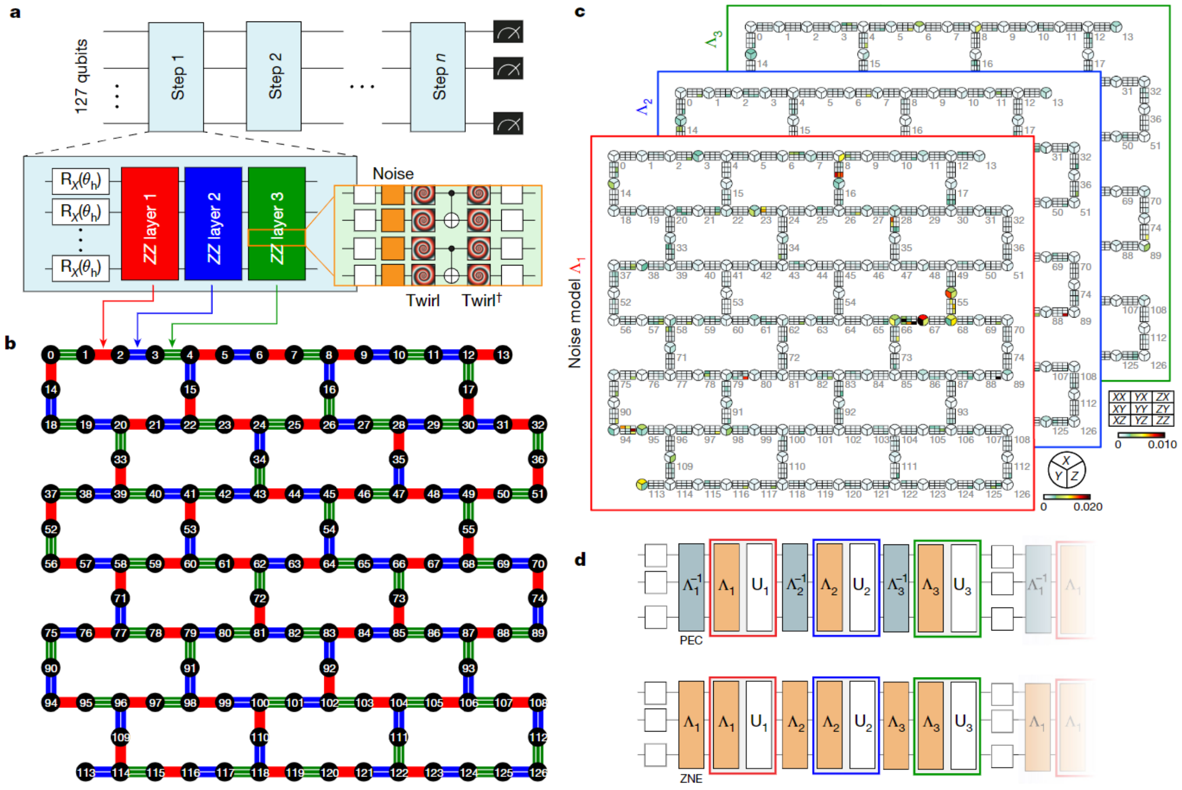
Temme, Bravyi, Gambetta, PRL **119**, 180509 (2017)
Berg, Minev, Kandala, Temme, Nature Physics **19**, 1116 (2023)

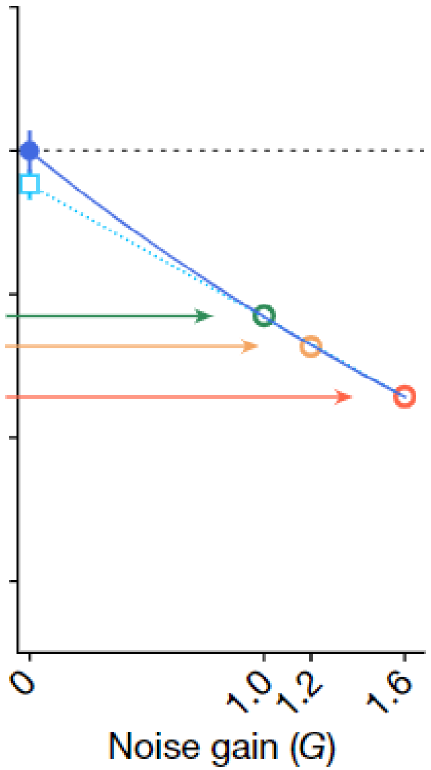
How do PEC and PER work?

$\boxed{\text{Multiply the average by } \gamma^{n_{\mathrm{CNOT}}}}$ $\Rightarrow$ **exponentially costly when reducing noise**

Possible solution: increase noise (avoid quasiprobability distributions!)

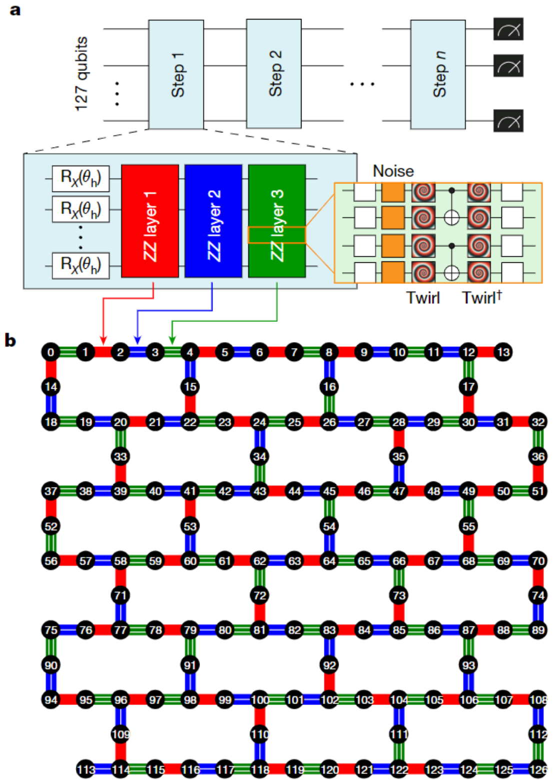
E.g., IBM "quantum utility" experiment:





Kim et al., Nature **618**, 500 (2023)

Quantum utility?



Kim et al., Nature **618**, 500 (2023)

SIGBOVIK 2024 proceedings, page 199

Quantum Disadvantage

Or, simulating IBM's 'quantum utility' experiment with a Commodore 64

Anonymous
quantum.disadvantage@proton.me
https://quantum-disadvantage.co.uk

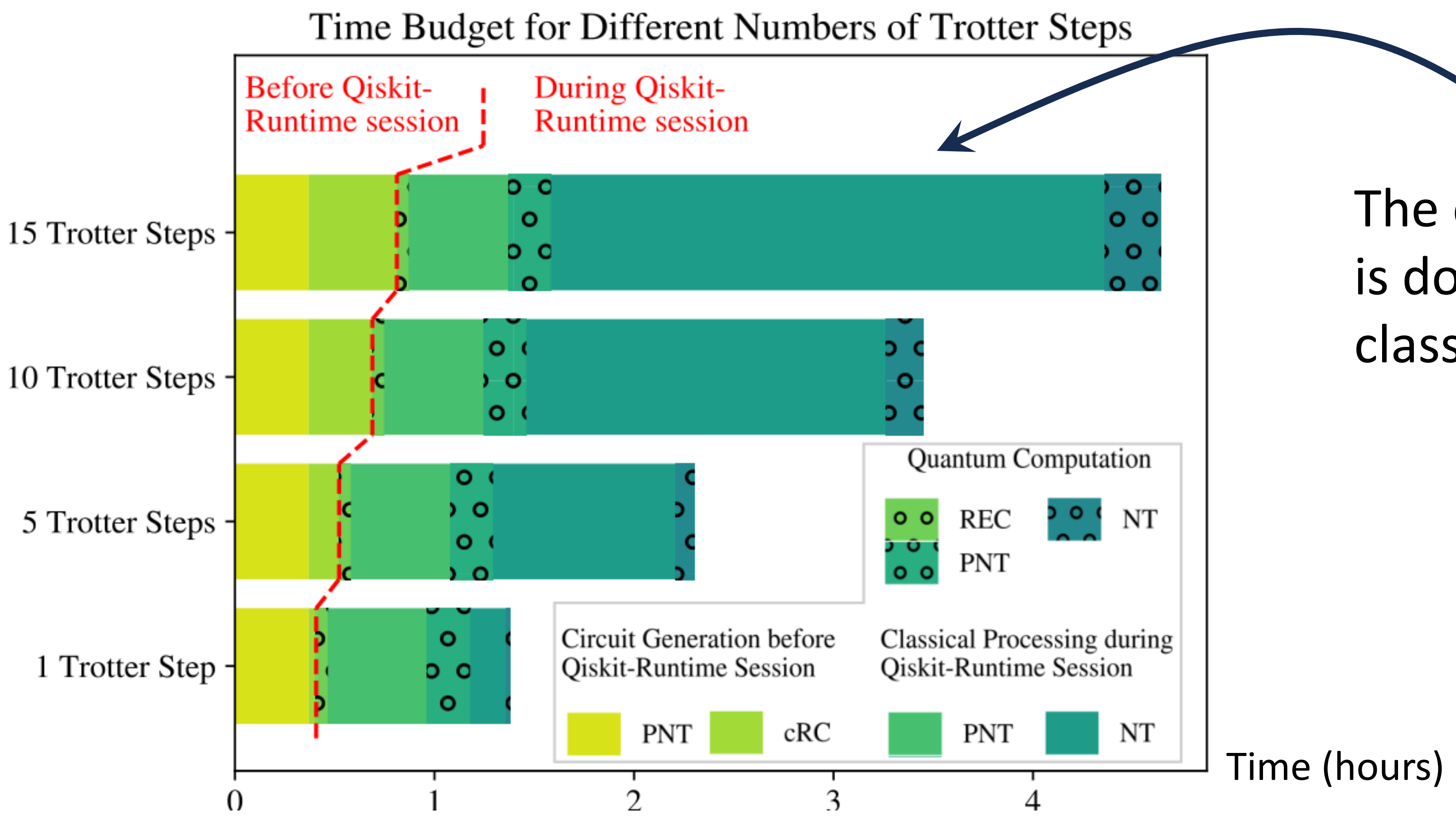
March 29, 2024

Abstract

measuring expectation values of a Trotterized Ising model on their 127-qubit
n that their results were unlikely to be replicable on a classical computer. In the
implement a classical simulation of the same experiment using a Commodore
)ulwich Quantum. Our simulation requires less than 15kB of memory and
ime per data point. To accomplish this we use a variant of the sparse Pauli
d by Begušić and Chan. We show that aggressive truncation combined with
ibitive (for a C64) memory cost of storing the truncated Pauli basis in SPD,
tch the error-mitigated results obtained from the quantum device.

Figure 2: The experimental setup – a Commodore 64 is connected to a monitor through a composite video to HDMI converter, with the code cartridge inserted into the expansion port.

Cost of increasing sampling?



The computation time is dominated by classical generation of trajectories

(Can be much improved with adjusting the way of storing circuits?)

Note: noise evolves in time: need to be fast enough

Perrin, Scoquart, KS unpublished

Solution to the portability issue: QLM/Qaptiva

FRANCE HYBRID HPCQUANTUM INITIATIVE

BACQ
MetriQs-France

Welcome to Qaptiva Access Portal

QLMaaS-1.9.1

Qaptiva Access is a software solution that extends myQLM to submit Quantum jobs to a Qaptiva Applicance: Qaptiva 800s Plugins and QPUs are available through myQLM.

Install | Get started | Configure

Qaptiva Access client is available on PyPI. This python library can be installed from your terminal using the following command:

```
python3 -m pip install myqlm
```

Useful links

Documentation | Display Notebooks | Download Notebooks

https://qlm35e.neasqc.eu/

Solution to the portability issue: QLM/Qaptiva

FRANCE HYBRID HPCQUANTUM INITIATIVE

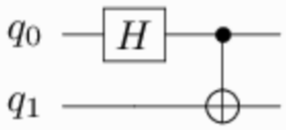
BACQ MetriQs 2030

Welcome to Qaptiva Access Portal

QLMaaS-1.9.1

Quantum Hello World

The following code snippet creates and simulates a simple Bell pair circuit:



Functional mode Sequential mode

```python
from qat.lang import qrout, H, CNOT

@qrout
def bell_pair():
    H(0)
    CNOT(0, 1)

result = bell_pair().run()

for sample in result:
    print(f"State {sample.state} amplitude {sample.amplitude}")
```

```
State |00> amplitude (0.7071067811865475+0j)
State |11> amplitude (0.7071067811865475+0j)
```

n jobs to a Qaptiva Applicance: Qaptiva 800s

Configure

led from your terminal using the following

Download Notebooks

https://qlm35e.neasqc.eu/

Solution to the portability issue: QLM/Qaptiva

FRANCE HYBRID HPCQUANTUM INITIATIVE

BACQ
MetriQs-France

Welcome to Qaptiva Access Portal

QLMaaS-1.9.1

Qaptiva Access is a software solution that extends myQLM to submit Quantum jobs to a Qaptiva Applicance: Qaptiva 800s Plugins and QPUs are available through myQLM.

Install | Get started | Configure

Qaptiva Access client is available on PyPI. This python library can be installed from your terminal using the following command:

```
python3 -m pip install myqlm
```

Useful links

Documentation

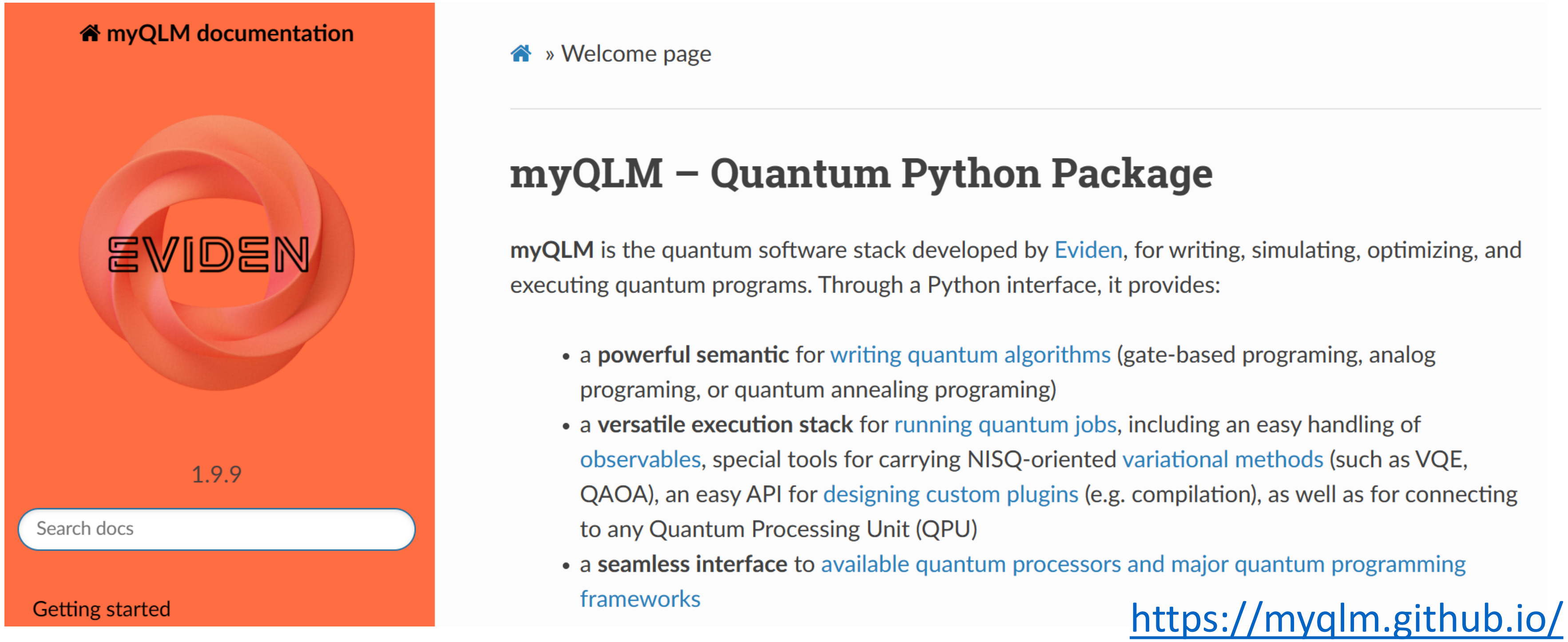
Display Notebooks

Download Notebooks

https://qlm35e.neasqc.eu/

Qaptiva (paid) — advanced simulation
+ (in the future) access to quantum hardware

Solution to the portability issue: QLM/Qaptiva



https://myqlm.github.io/

Solution to the portability issue: QLM/Qaptiva

FRANCE HYBRID HPCQUANTUM INITIATIVE

BACQ MetriQs-France

myQLM documentation

EVIDEN

» Welcome page

myQLM – Quantum Python Package

myQLM is the quantum software stack developed by Eviden, for writing, simulating, optimizing, and executing quantum programs. Through a Python interface, it provides:

- a **powerful semantic** for writing quantum algorithms (gate-based programing, analog programing, or quantum annealing programing)
- a **versatile execution stack** for running quantum jobs, including an easy handling of observables, special tools for carrying NISQ-oriented variational methods (such as VQE, QAOA), an easy API for designing custom plugins (e.g. compilation), as well as for connecting any Quantum Processing Unit (QPU)
- a **seamless interface** to available quantum processors and major quantum programming frameworks

https://myqlm.github.io/

Interoperability

Warning

Interoperability packages are deprecated for Python versions 3.6 and 3.7

This package enables access to other quantum programming environments such as Qiskit, ProjectQ, PyQuil, Cirq ... This package will not automatically install dependency packages because someone who want to interface with Qiskit may not want to interface with Cirq... The desired quantum environment can be cherry-picked using the pip command:

Qiskit | ProjectQ | Cirq | PyQuil | All frameworks

```
pip install myqlm-interop[qiskit_binder]
```

myQLM = my Quantum Learning Machine

myQLM (free) — gate-based programming and noiseless simulation

Lots of examples