SAND2022-12744C

Sandia National Laboratories

High-level benchmarks and low-level characterization

PRESENTED BY

Timothy Proctor

Quantum Performance Laboratory
@ Sandia National Laboratories
Livermore, CA and Albuquerque, NM, (and Chicago, Il)

U.S. DEPARTMENT OF ENERGY | NNSA

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Quantum Performance Laboratory

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Outline

1. Holistic benchmarking

- Volumetric benchmarking
- Algorithmic benchmarks

2. Detailed error characterization

- State and process tomography
- Gate set tomography

Outline

1. Holistic benchmarking

- Volumetric benchmarking
- Algorithmic benchmarks

Many methods sit between these two extremes

- One/two-qubit RB[1]
- Cycle benchmarking[2]
- Pauli noise learning[3]
- ….

2. Detailed error characterization

- State and process tomography
- Gate set tomography

<u>High-level performance assessment</u>
Aims to answer questions like:

- What's this device's failure rate when running many-qubit circuits?
- What algorithms can this device run?
- Is device X or device Y better?

<u>Low-level performance assessment</u>
Aims to answer questions like:

- What kinds of errors are occurring?
- What's the error rate of each kind of fundamental error on each gate?
- What's the process fidelity of this gate?

[1]E. Magesan et al, , Phys. Rev. Lett. 106, 180504 (2011), [2]A. Erhard et al, Nat. Commun. 10, 5347 (2019). [3]R. Harper et al, Nat. Phys. 16, 1184 (2020).

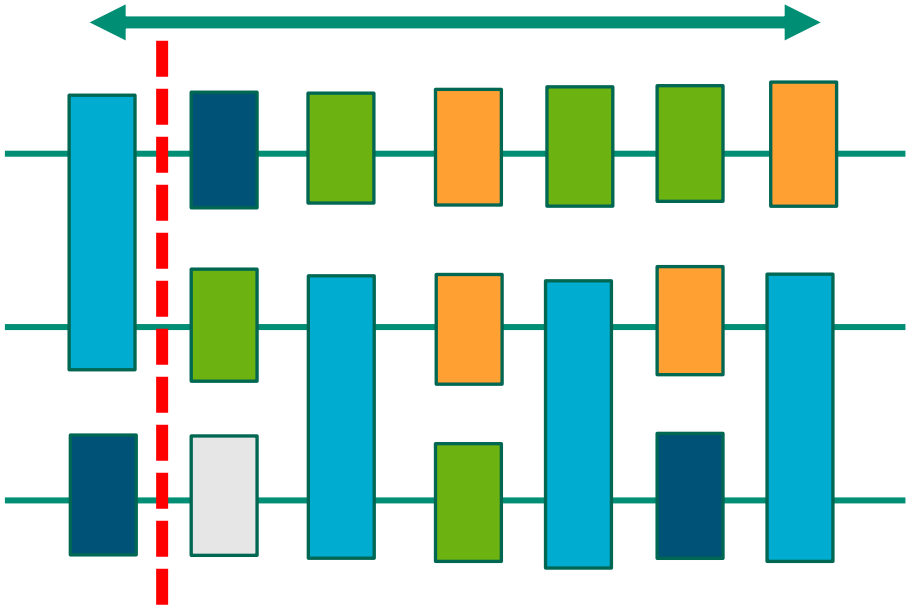
Recap: what is a quantum circuit?

Circuit Depth

d layers ("cycles") of logic gates

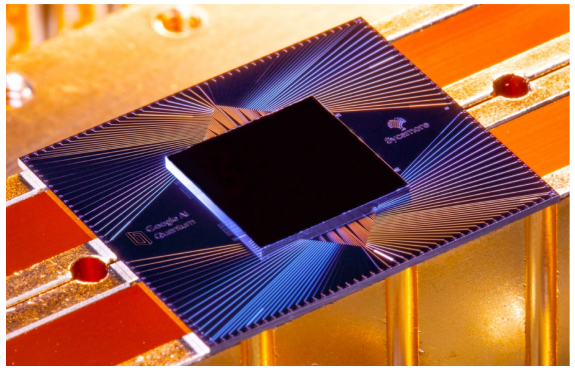
Circuit Width

w qubits



Many methods assume implicit "barriers" between every layer that prevent *compilation*, i.e., collapsing the circuit to a simpler circuit that implements the same unitary.

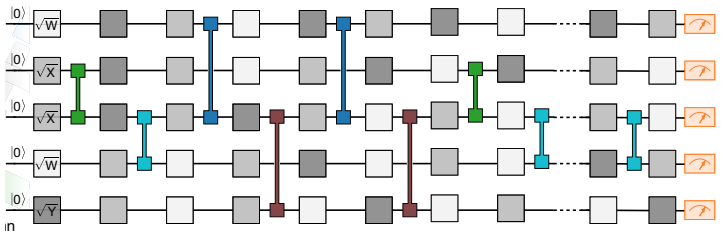
Holistic benchmarking



Arute et al., Nature 574, 505 (2019)

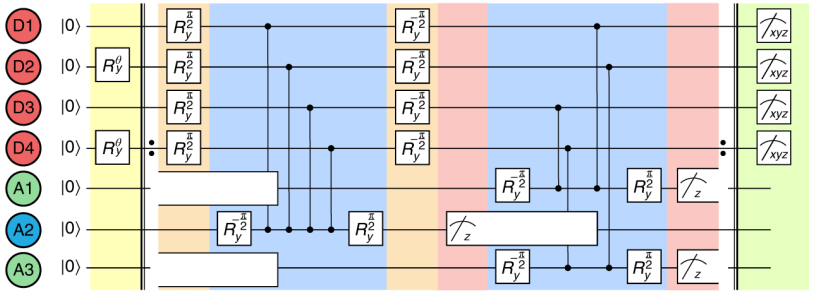
What applications or algorithms can it run?

Can it run random circuit sampling?



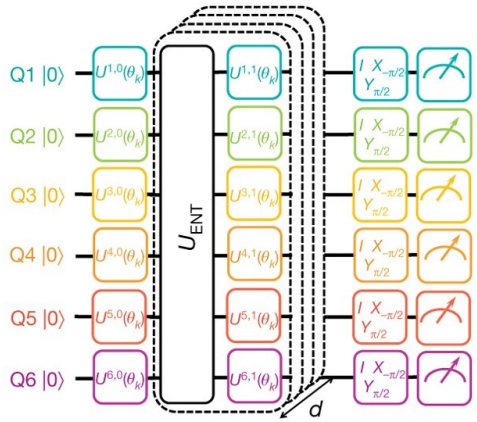
Arute et al., Nature 574, 505 (2019)

Quantum error correction?

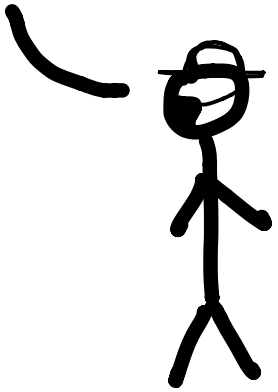


Anderson et al., Nat. Phys. 16, 875 (2020)

What about QAOA?



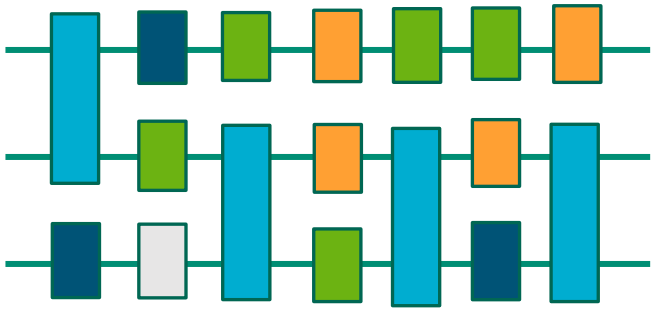
Kandala et al.,Nature 549, 242 (2017)



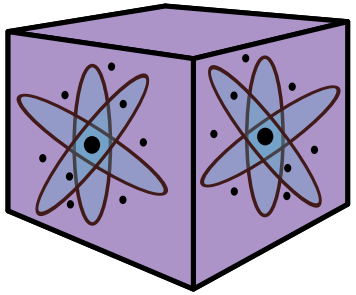
What about VQE? What about Phase Estimation? The QFT? Etc… etc…

Noisy quantum computers

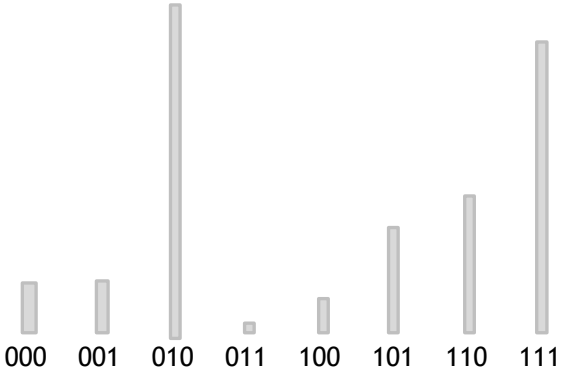
My quantum circuit



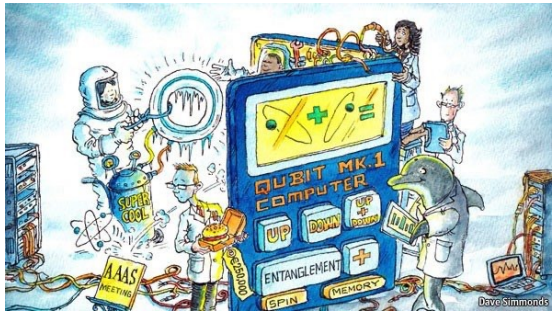
My (imaginary) perfect quantum computer

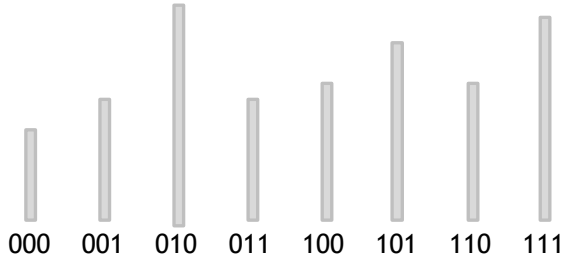


Distribution over bit strings

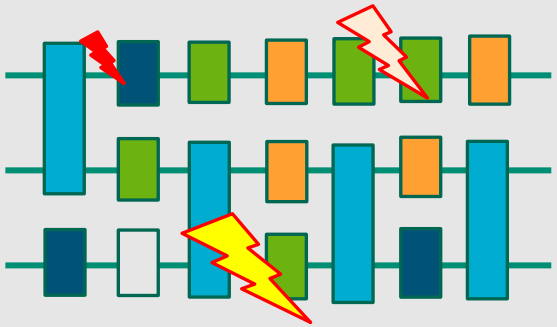


A real quantum computer





Errors occur when circuits are run on real quantum computers!



"Success probability" = "Distance between these distributions"

The quantum volume benchmark

- The quantum volume benchmark[1] is based on a particular family of random circuits:

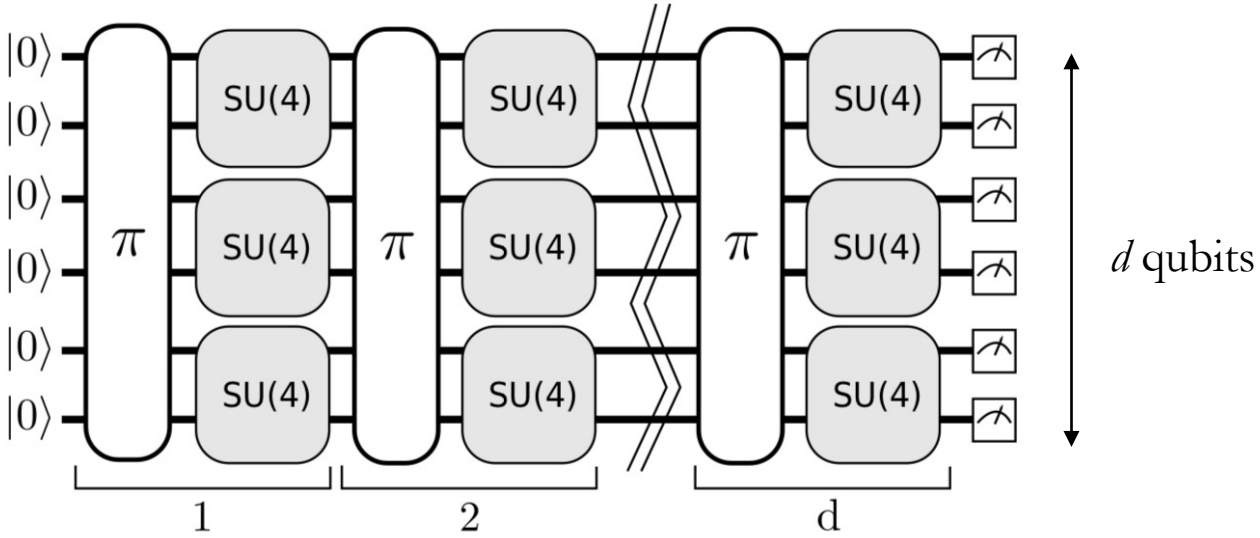


Image from Cross et al., PRA 100, 032328 (2019)

- The quantum volume benchmark is a "full stack" benchmark: it permits **compilation**: you're only required to implement approximately the same unitary as the circuit.

- Therefore quantum volume does not isolate gate performance: it quantifies a complex combination of gate performance, compiler performance, and programmability limitations.

[1]Cross et al., PRA 100, 032328 (2019)

Beyond square circuits: volumetric benchmarking with quantum volume's circuits

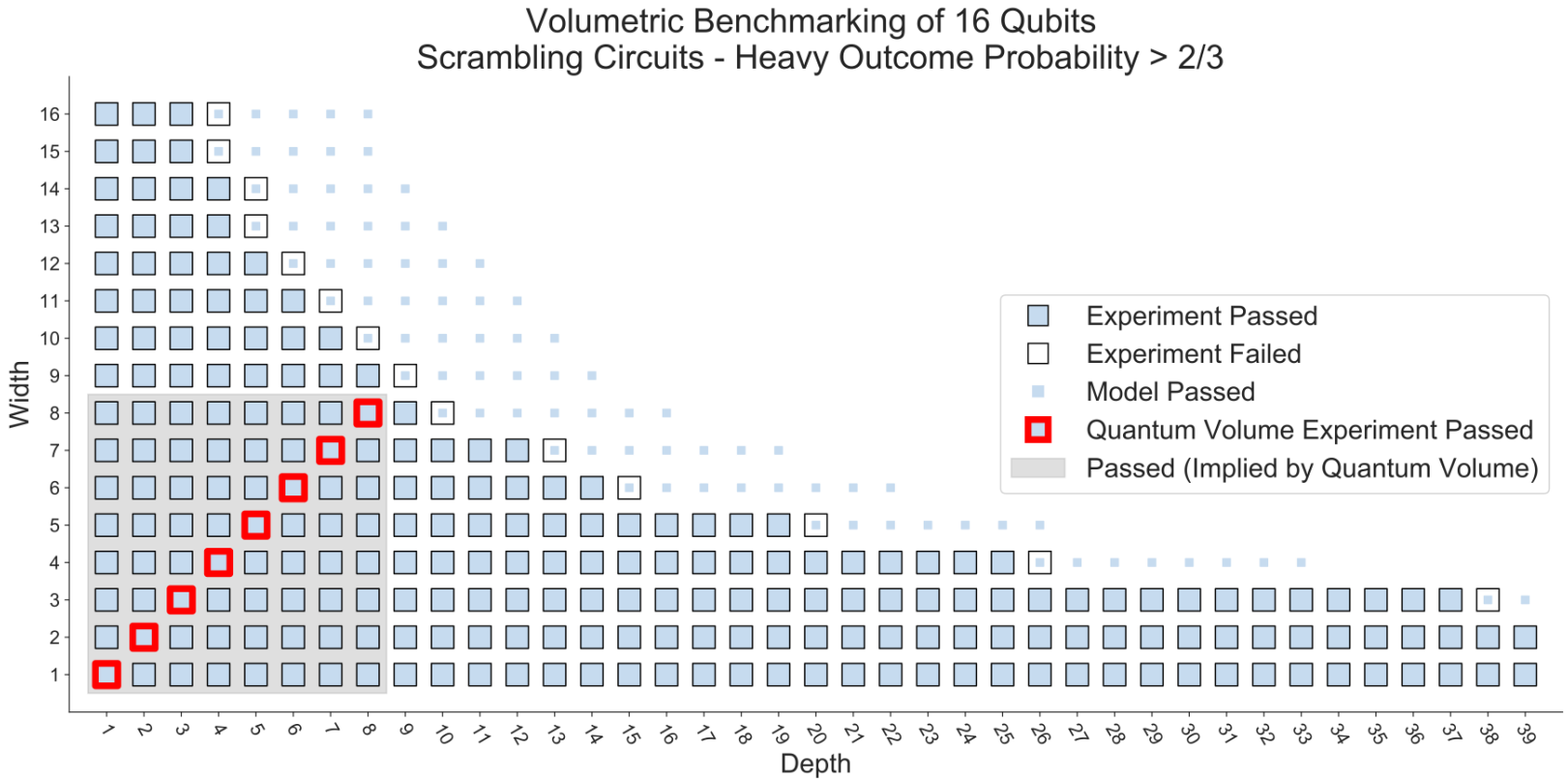
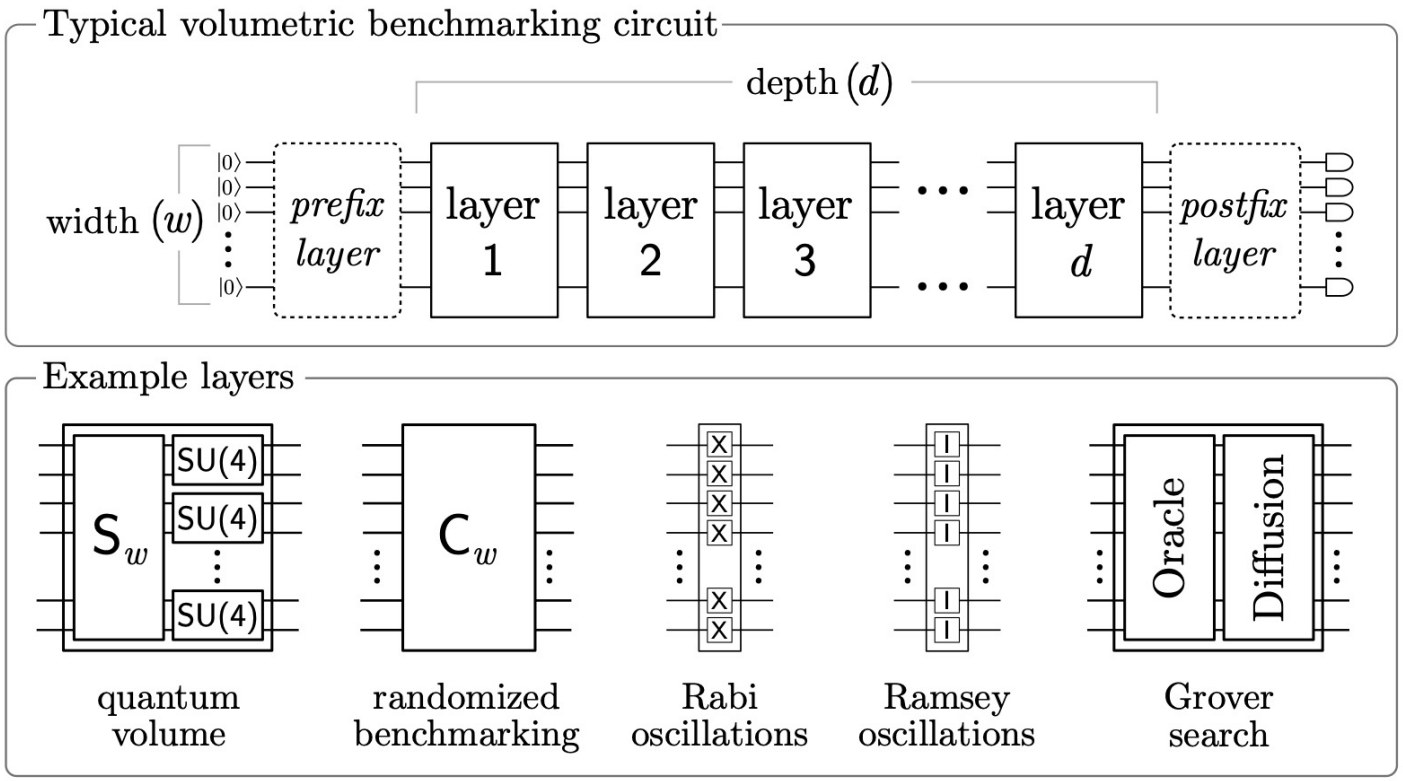
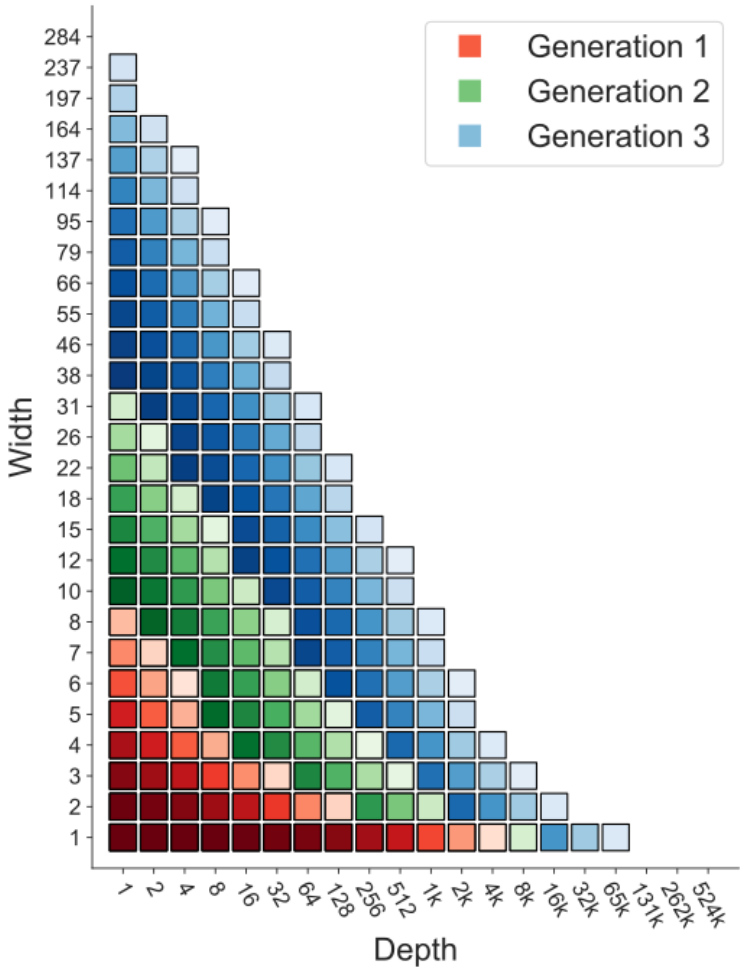


Figure 8(a). Volumetric benchmarking of a 16 qubit device using scrambling circuits. If at least 2/3 of the measurement results are heavy for a given width/depth pair, then the pair passes the test and is marked with a large, solid blue box. Using linear axes, the quantum volume experiments appear along the diagonal and are outlined with heavy, red lines. For this example, $\log_2(V_Q) = 8$. It is expected that scrambling circuits with both width and depth less than or equal to the quantum volume should succeed, and we highlight these with a gray background.

Blume-Kohout and Young, Quantum 4, 362 (2020)

Beyond the QV circuits: general volumetric benchmarking



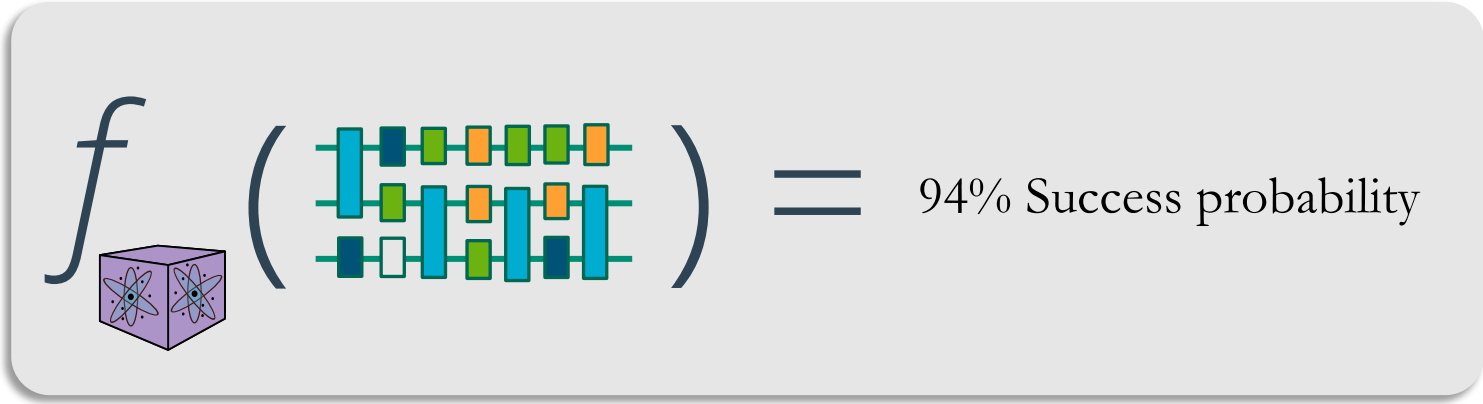


Almost any family of circuits can be used for volumetric benchmarking!

Blume-Kohout and Young, Quantum 4, 362 (2020)

Holistic benchmarking aims to probe a quantum computer's "capability"

- We define a quantum computer's *capability* to be a map from the space of circuits to the circuit's success rate (quantified, e.g., by fidelity).

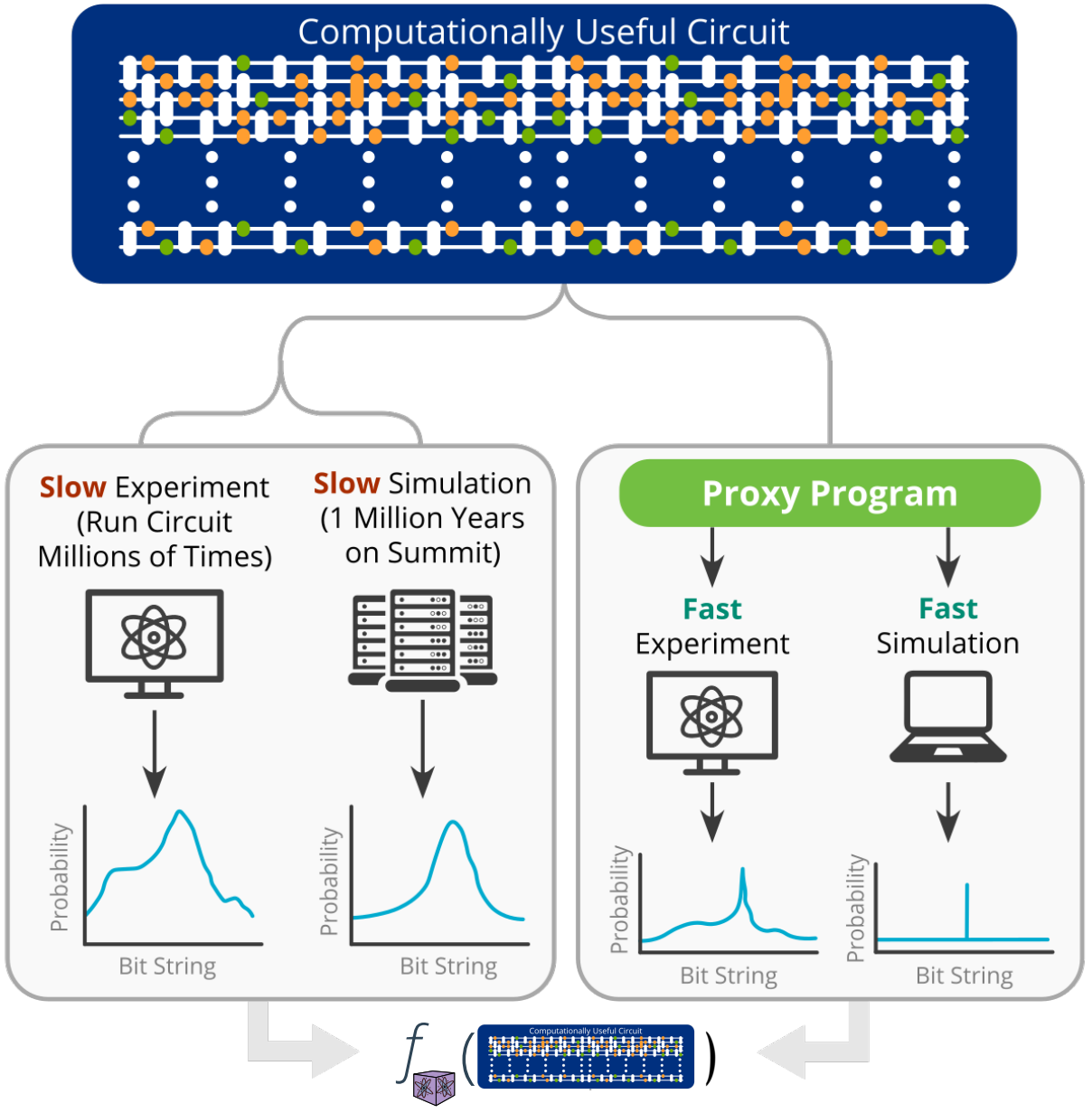


- This function is what (most) holistic benchmarks are trying to probe and understand.

- But we can't just measure f(C) using an experiment…

The long-term challenge: avoiding exponential resource scaling

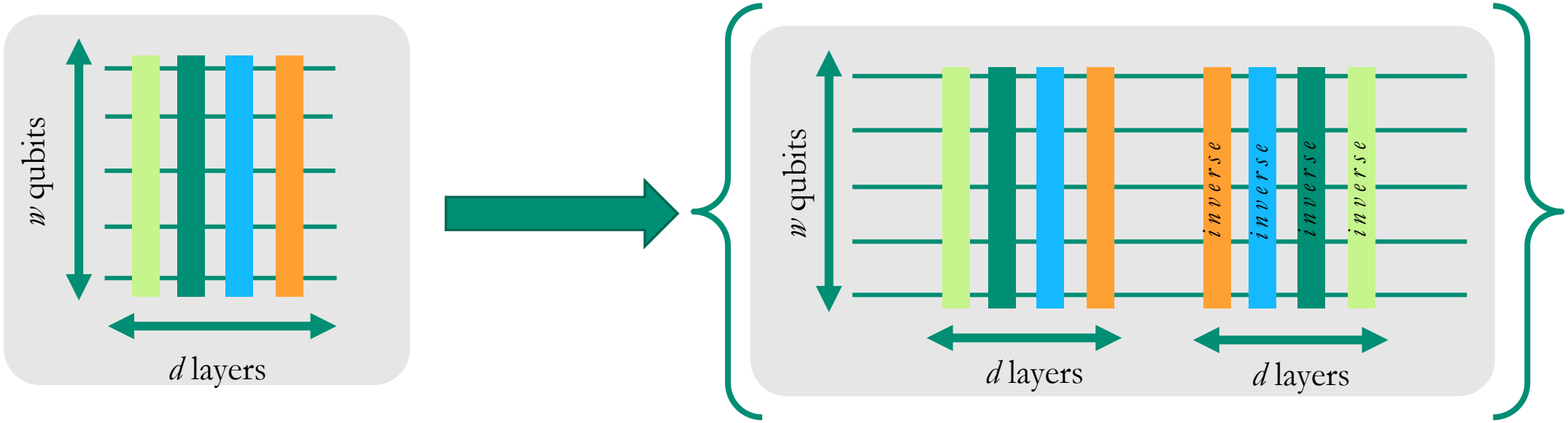
The obvious approach to benchmarking



Instead we must use carefully designed, efficiently verifiable "proxy programs"

Mirror circuit benchmarks

One solution – converting any circuit C to a set of "mirror circuits":

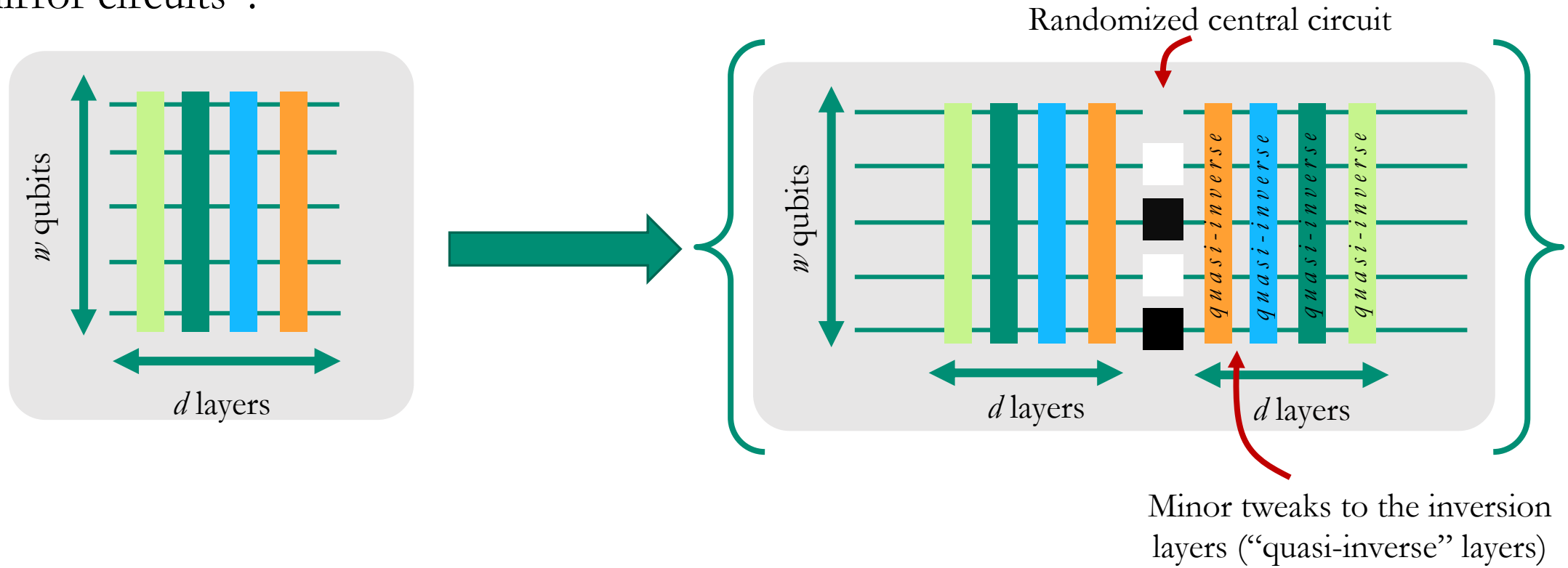


Flawed due to error cancelation/addition!

Proctor et al, Nat. Phys. 18, 75-79 (2022)

Mirror circuit benchmarks

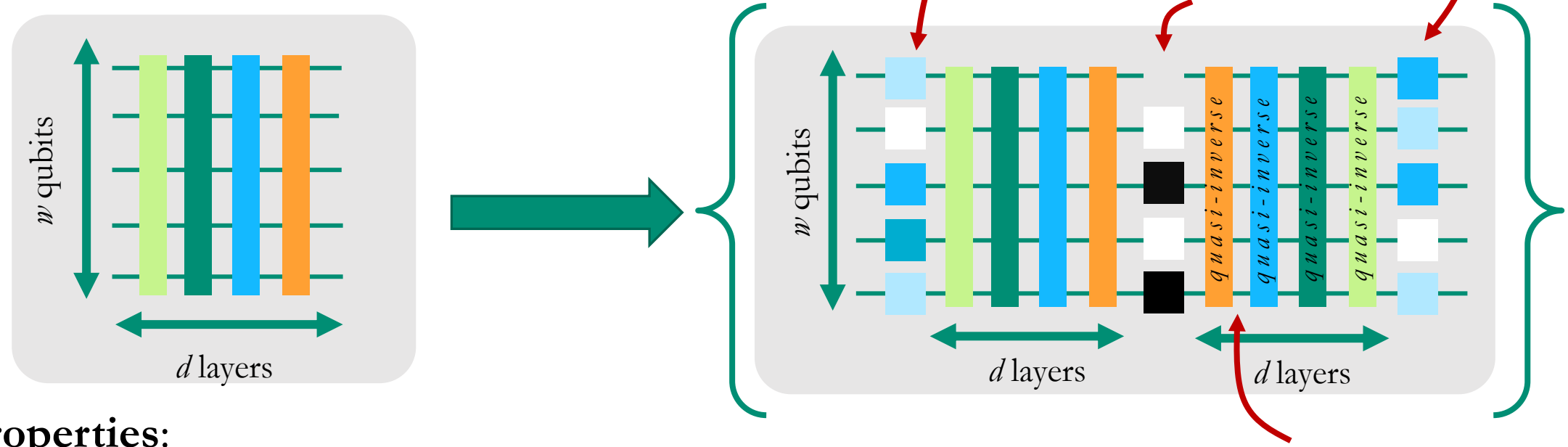
One solution – converting any circuit C to a set of "mirror circuits":



Proctor et al, Nat. Phys. 18, 75-79 (2022)

Mirror circuit benchmarks

One solution – converting any circuit C to a set of "mirror circuits":



Properties:

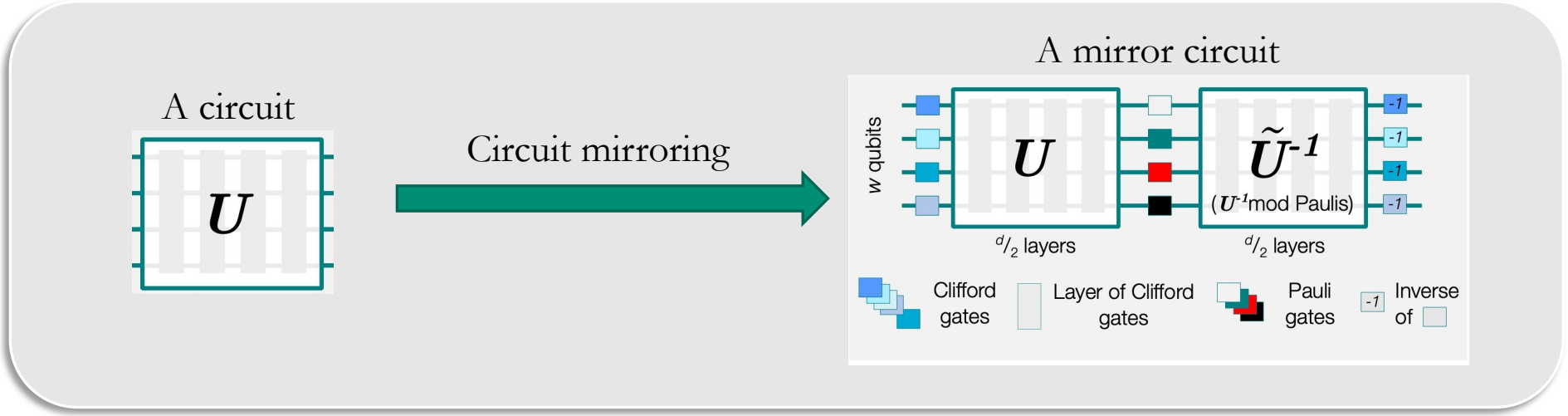
1. Every mirror circuit has an efficiently verifiable output.
2. A processor's performance on these mirror circuits is representative of how well it will perform on C.

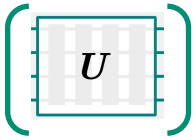
These circuit's success probabilities are related to fidelity of C!

Proctor et al, Nat. Phys. 18, 75-79 (2022)

Creating scalable and flexible benchmarks using mirror circuits

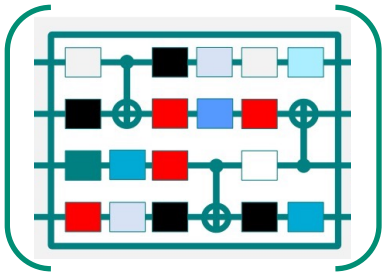
Circuit mirroring is a mapping that needs input circuit(s) to create a benchmark.



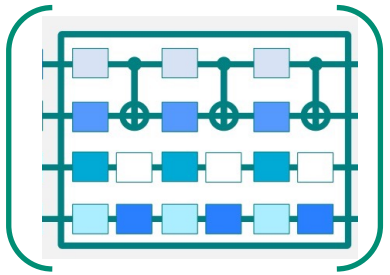
What circuits  create useful benchmarks?

Generic benchmarks

- Randomized, unstructured circuits

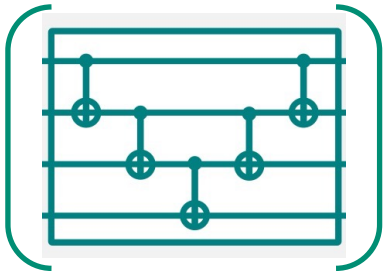


- Repetitive error-amplifying circuits



Application-specific benchmarks

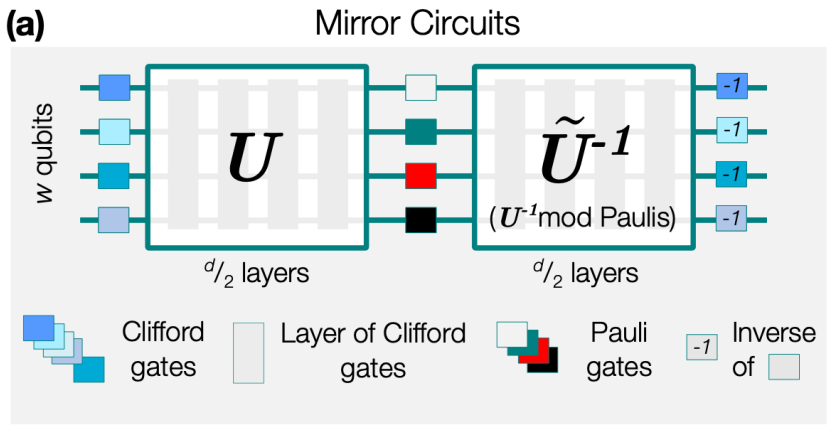
- An algorithm's circuit(s).

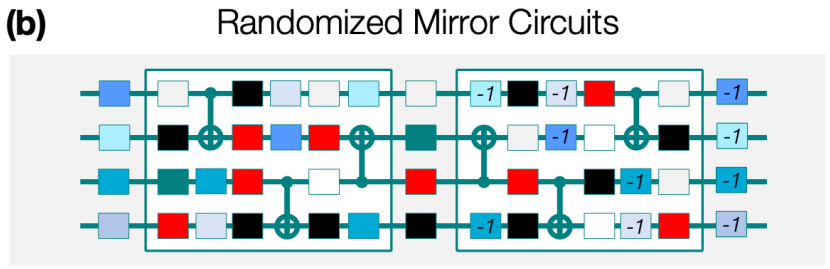


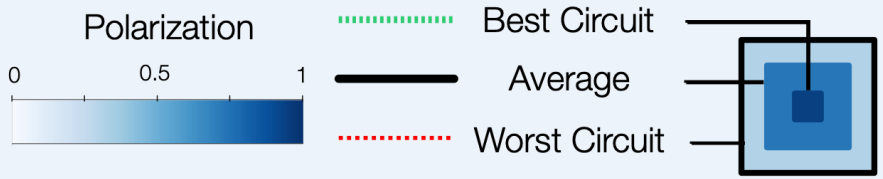
Proctor et al, Nat. Phys. 18, 75-79 (2022)

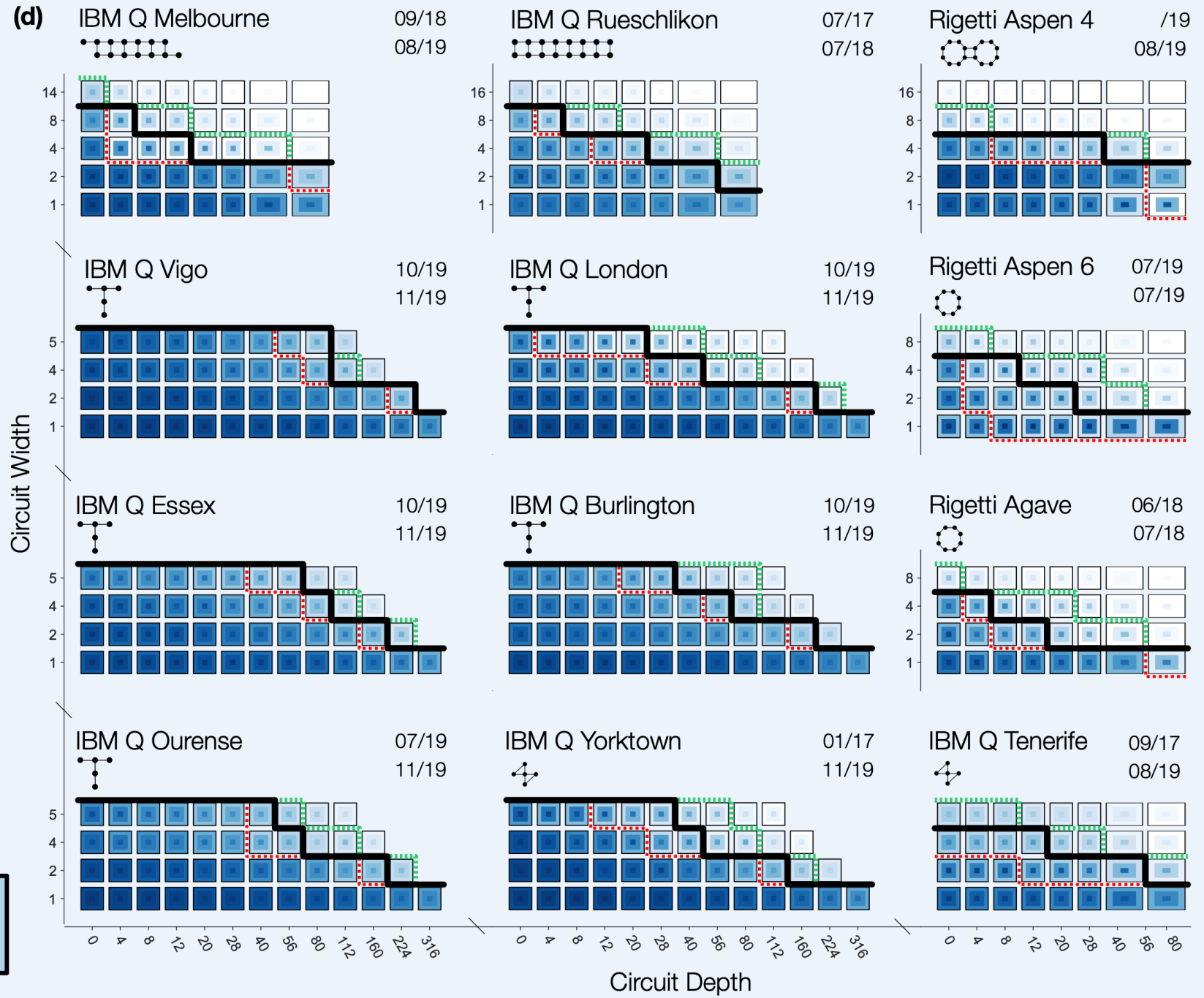
Experimental demo

Benchmarking 12 publicly accessible processors with randomized mirror circuits.





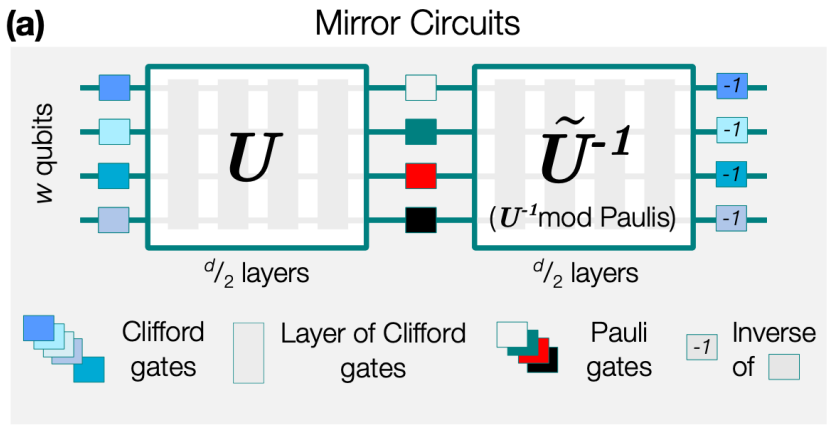


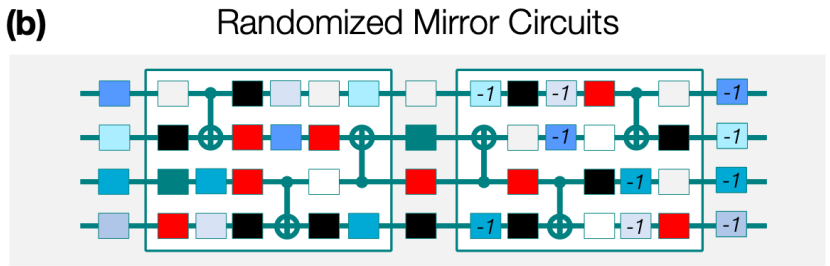


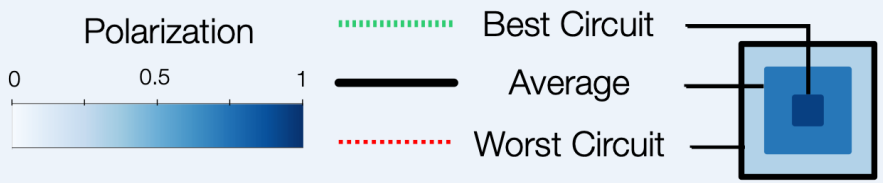
Proctor et al, Nat. Phys. 18, 75-79 (2022)

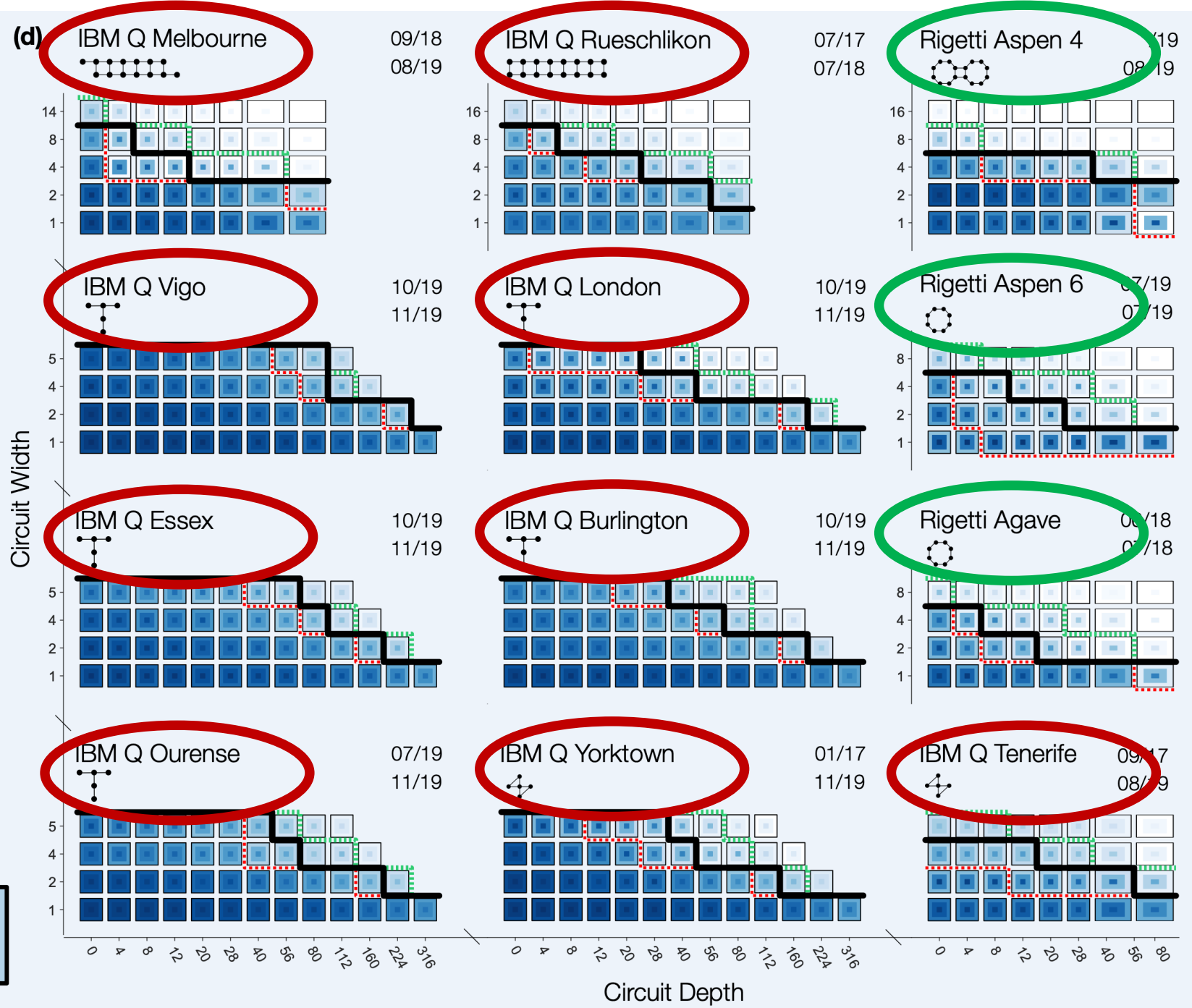
Experimental demo

The processors are from **IBM** and **Rigetti Computing.**







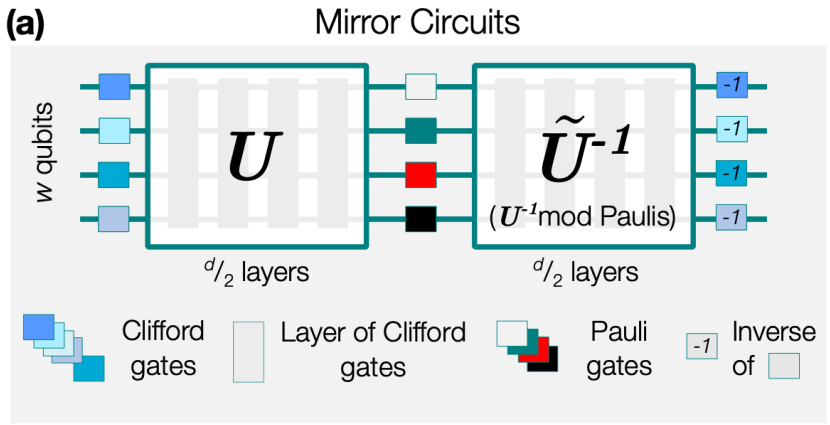


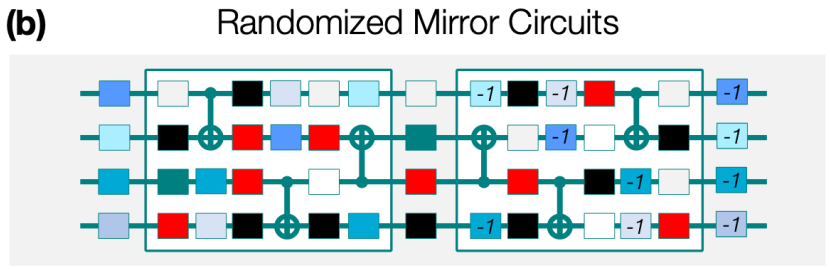
Proctor et al, Nat. Phys. 18, 75-79 (2022)

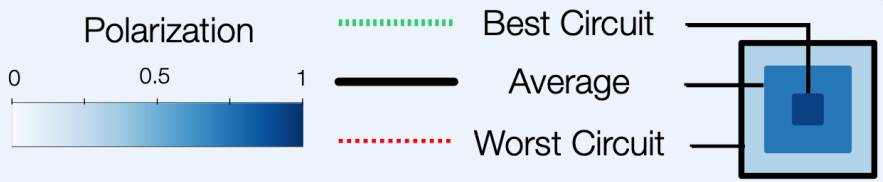
Experimental demo

We ran varied width and depth randomized mirror circuits, with approximately exponentially spaced **widths** and **depths**.

This is volumetric benchmarking.



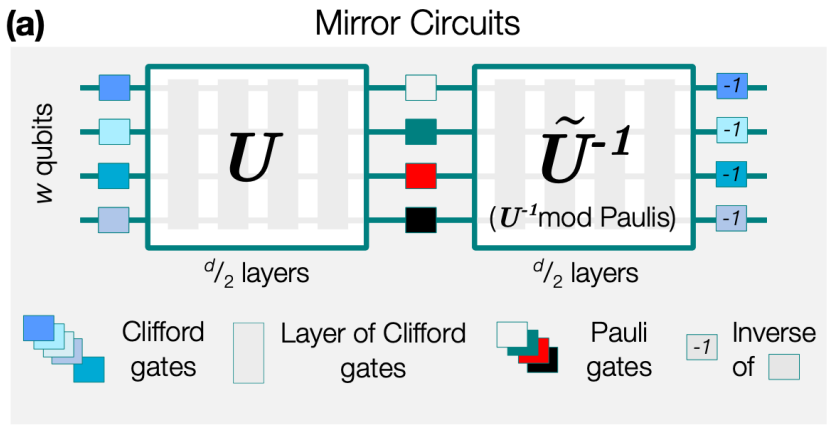


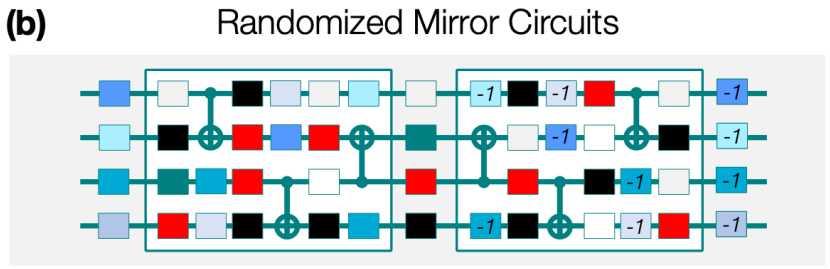


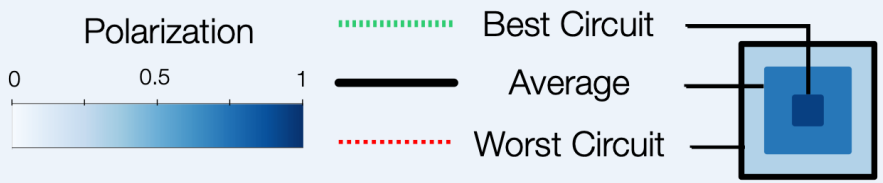
Proctor et al, Nat. Phys. 18, 75-79 (2022)

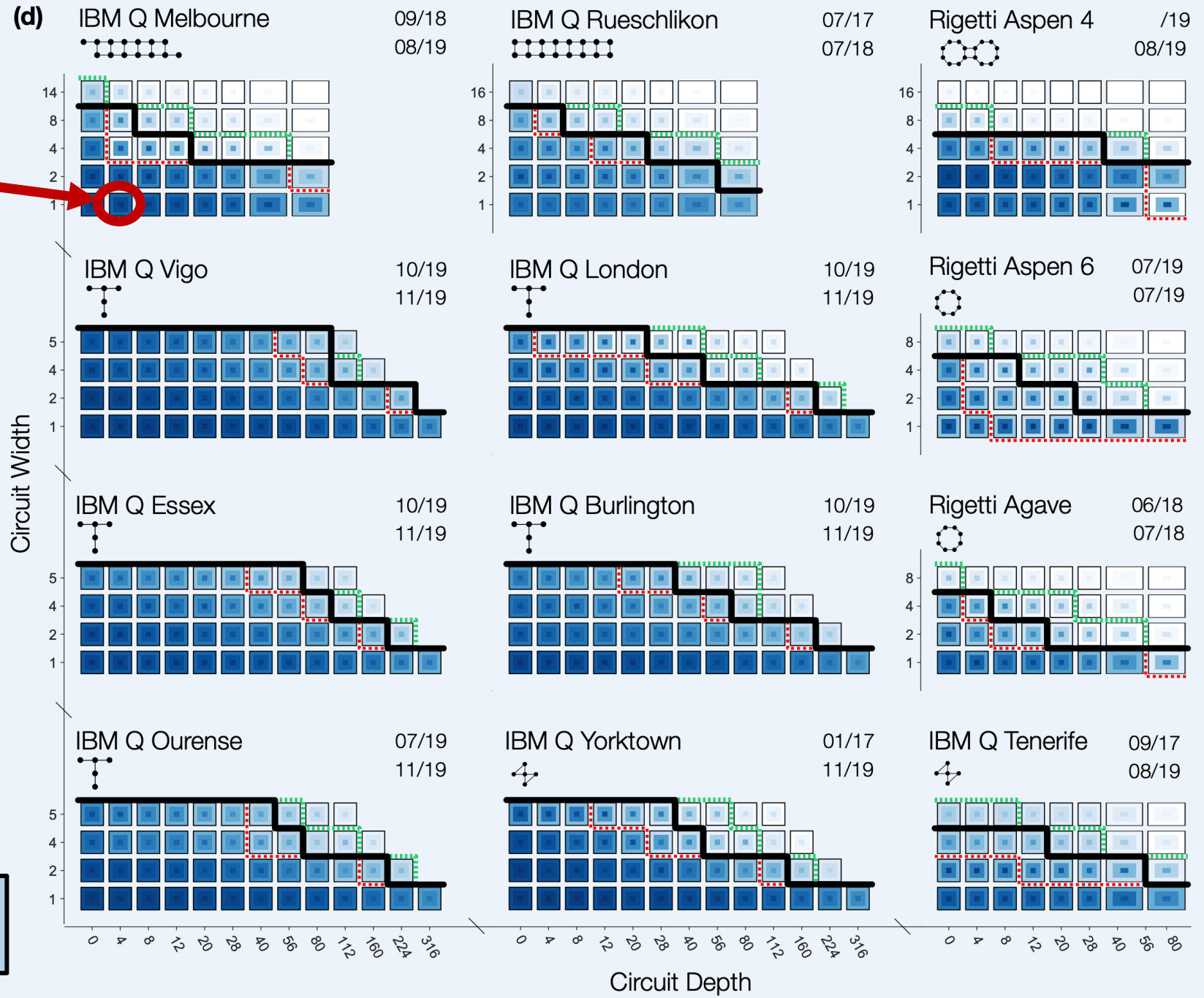
Experimental demo

At each (width, depth) pair we sampled 40 randomized mirror circuits, and we ran each one ~1000 times.





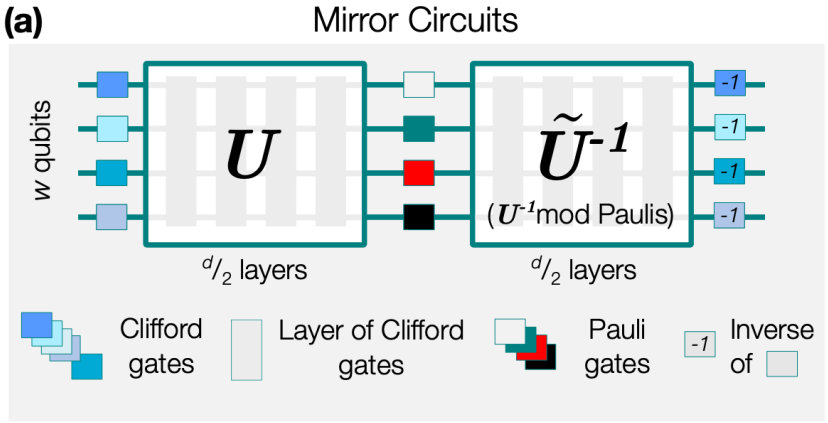


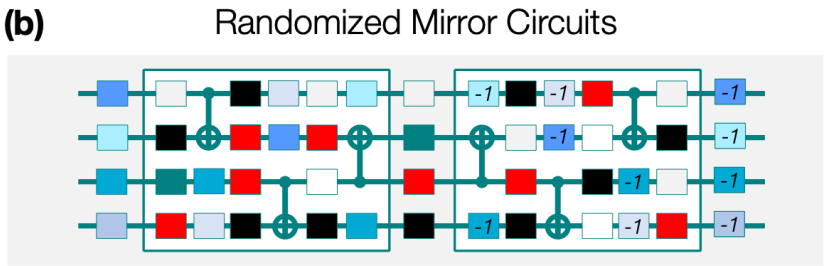


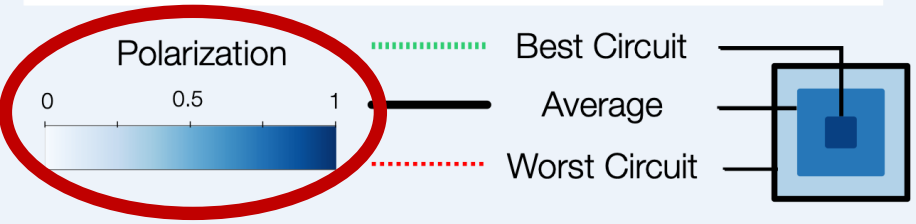
Proctor et al, Nat. Phys. 18, 75-79 (2022)

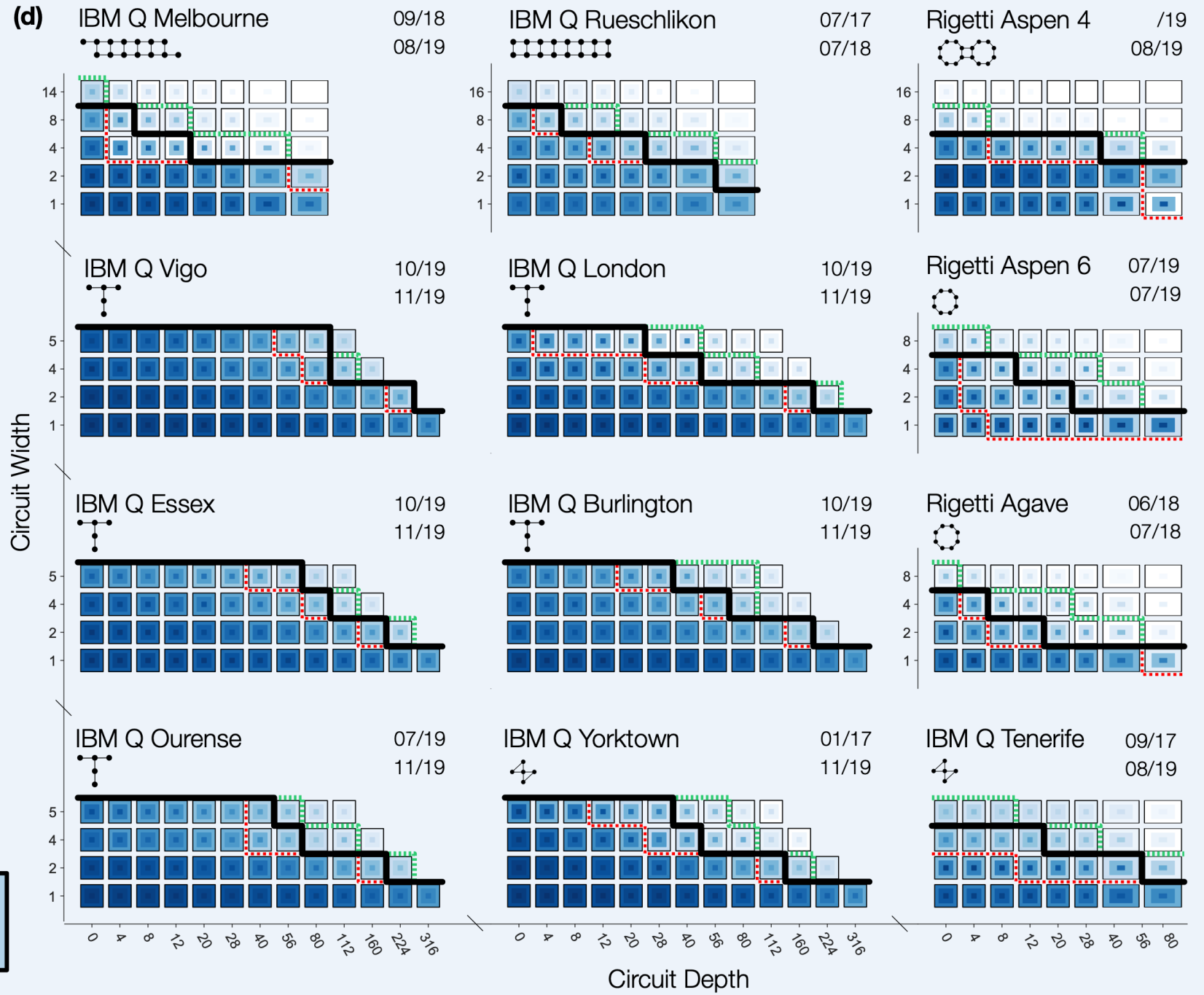
Experimental demo

We summarize performance via "polarization" = $(P - 1/2^w) / (1 - 1/2^w)$ where P is the probability that the correct bit-string is output by circuit, and w is it's width.





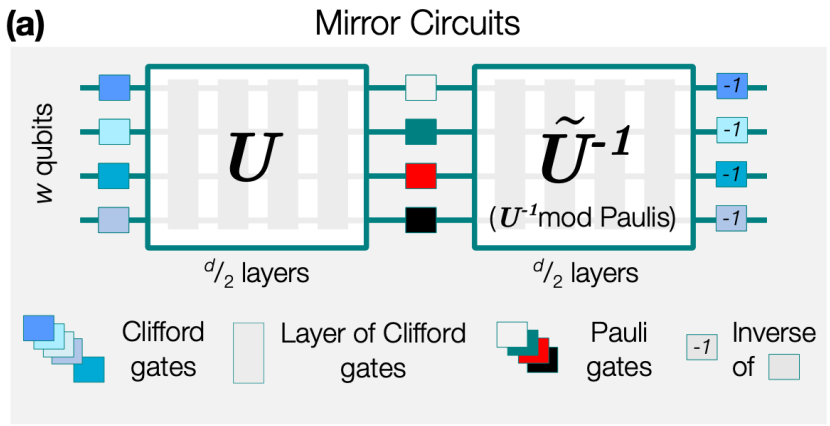


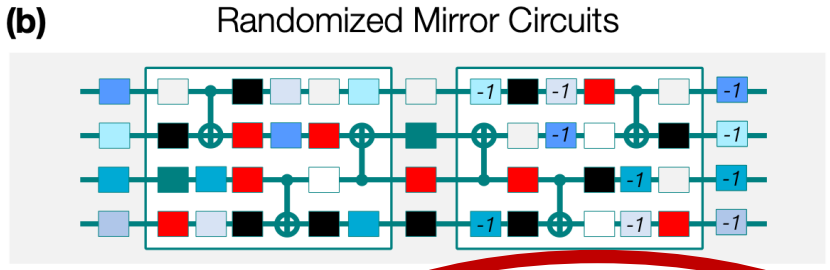


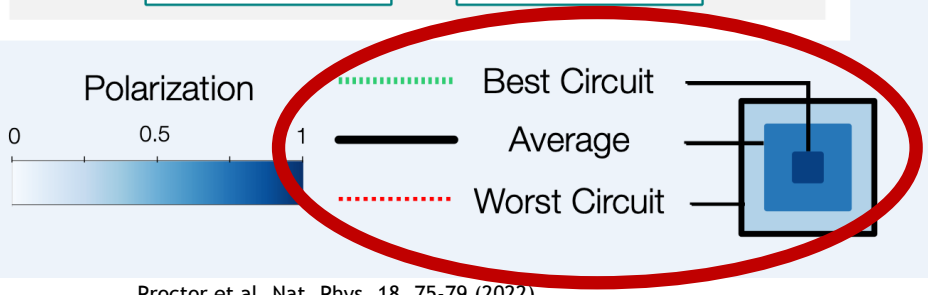
Proctor et al, Nat. Phys. 18, 75-79 (2022)

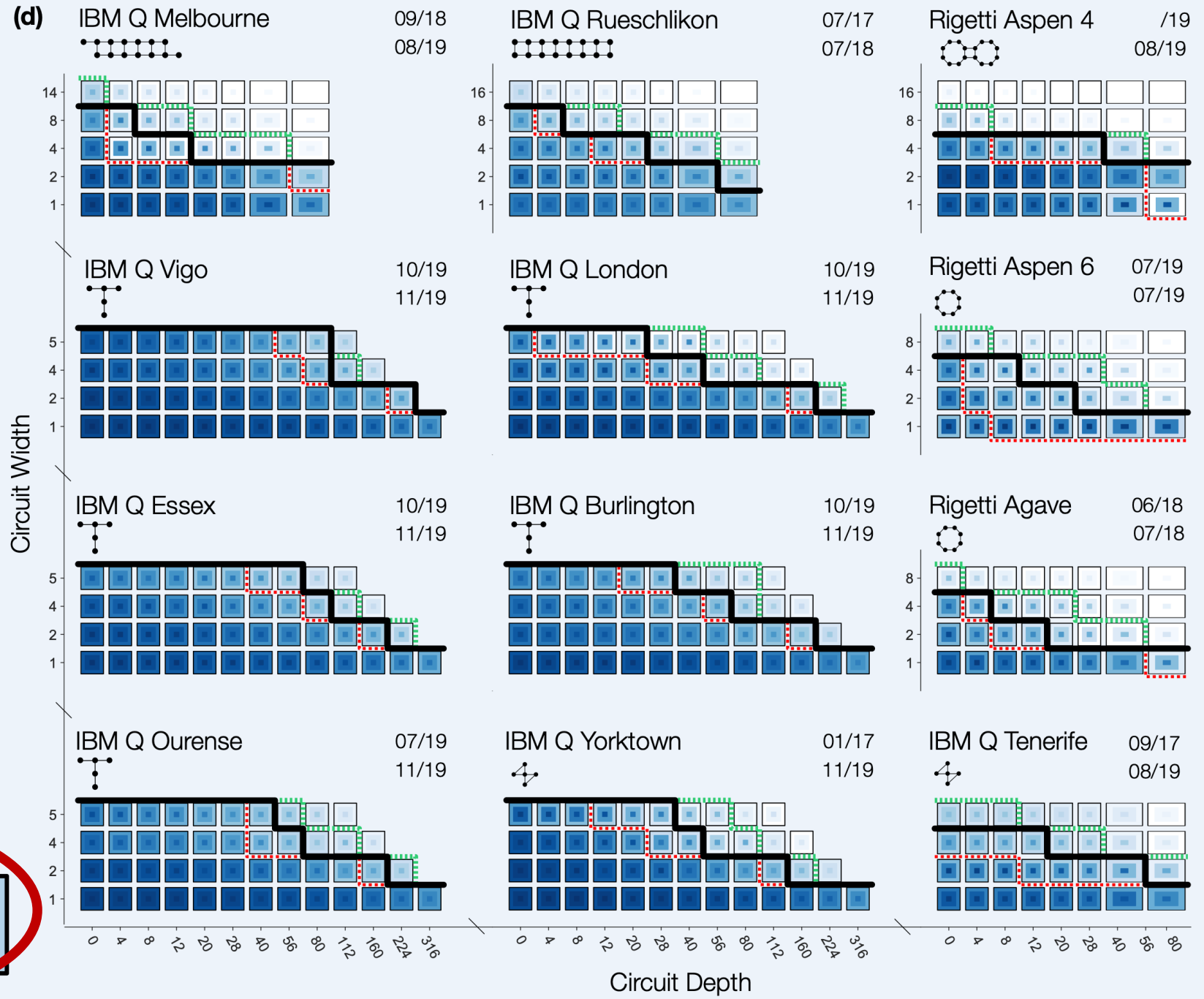
Experimental demo

We plot data for the best and worst performing of the 40 circuits – at each width and depth – as well as the mean performance.







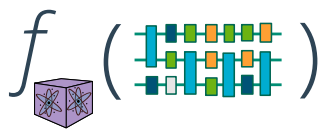


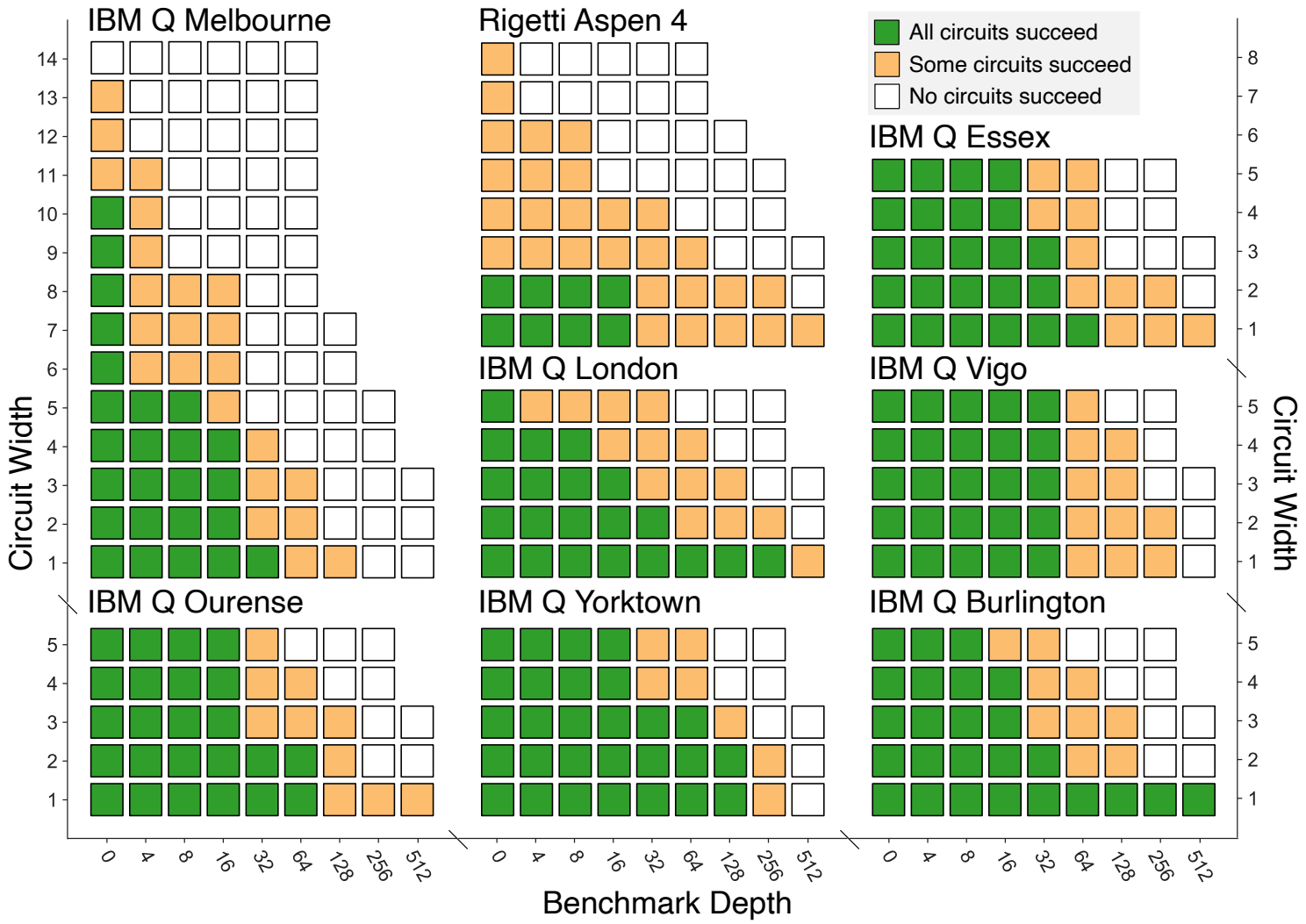
Proctor et al, Nat. Phys. 18, 75-79 (2022)

Capability regions

- The regions shown here include those circuit shapes where **all** / **some** / **none** of the test circuits within it succeeded with a polarization of at least 1/e.

- This can be thought of as a simple representation of the capability function

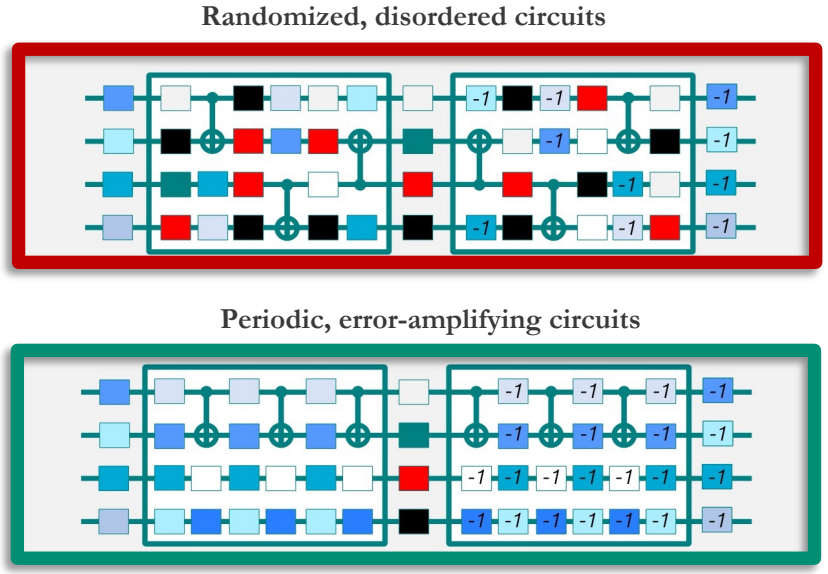


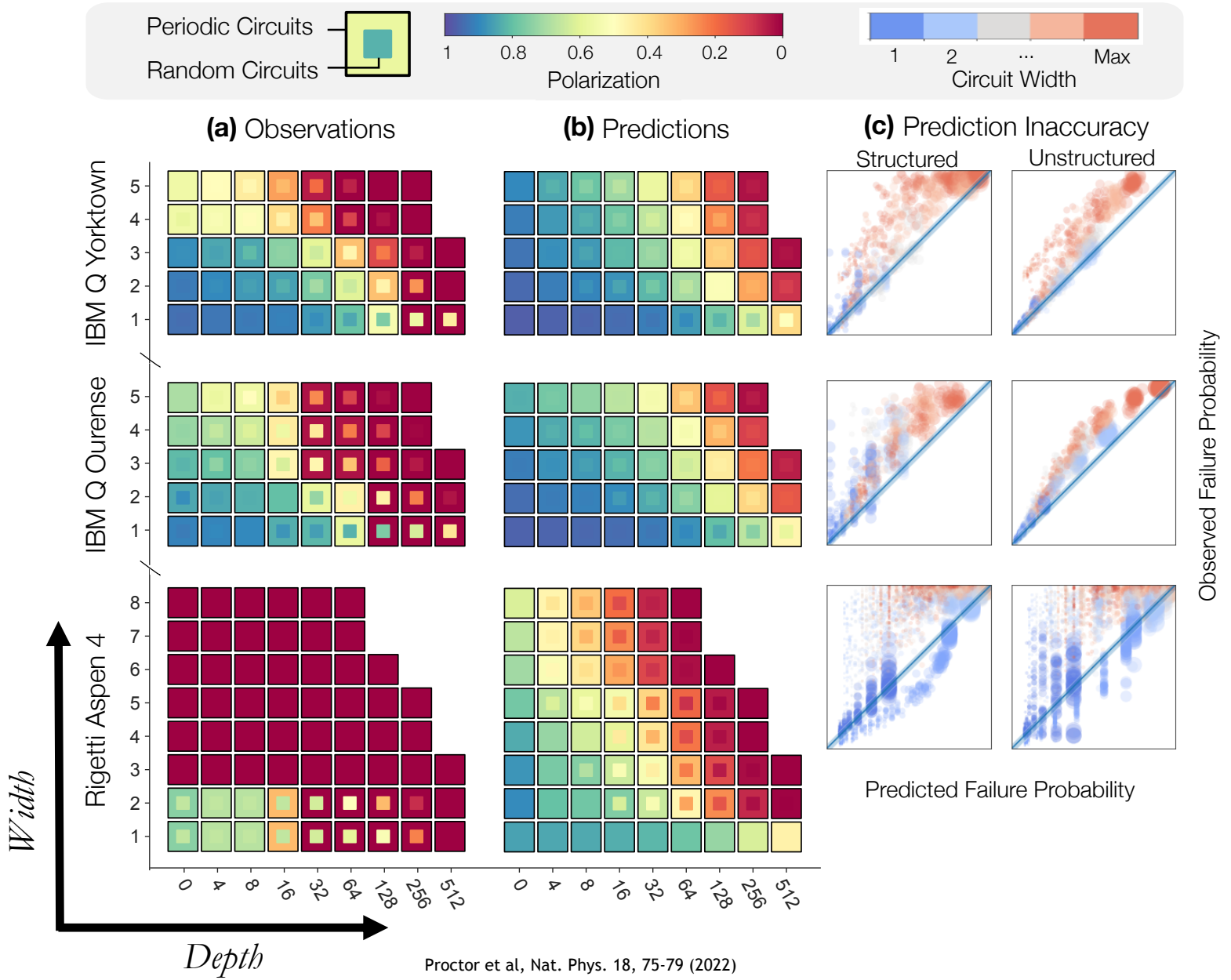


Proctor et al, Nat. Phys. 18, 75-79 (2022)

Do we need anything other than randomized benchmarks? Yes!

- Performance is typically **much** worse for periodic circuits!
- This is a symptom of **structured errors** that *cannot* be modelled by standard error rates.

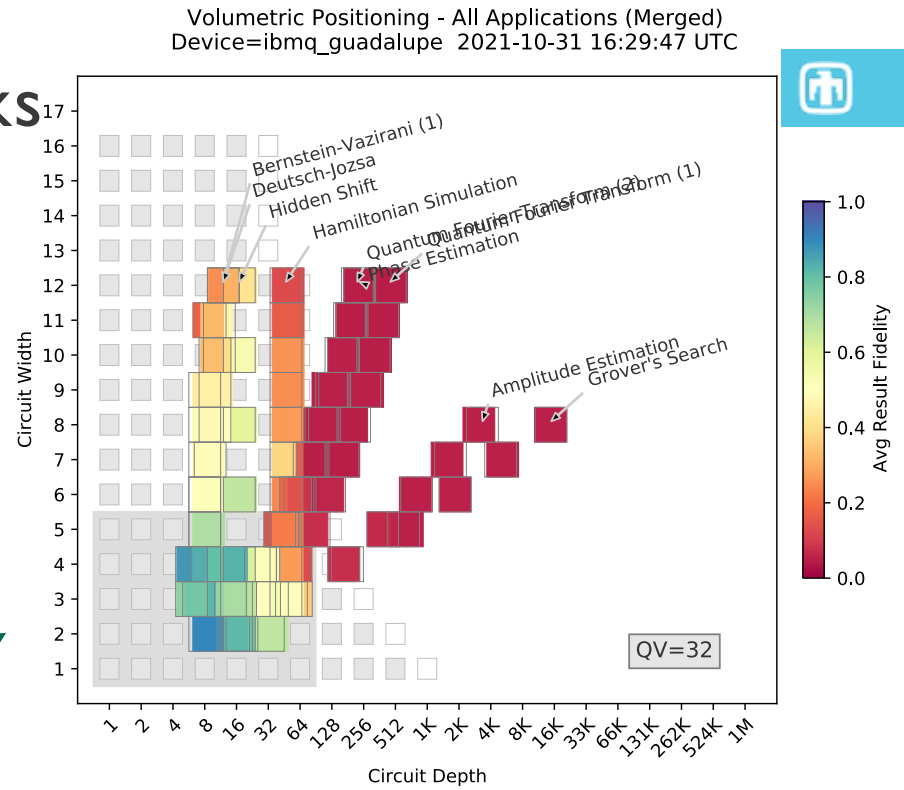


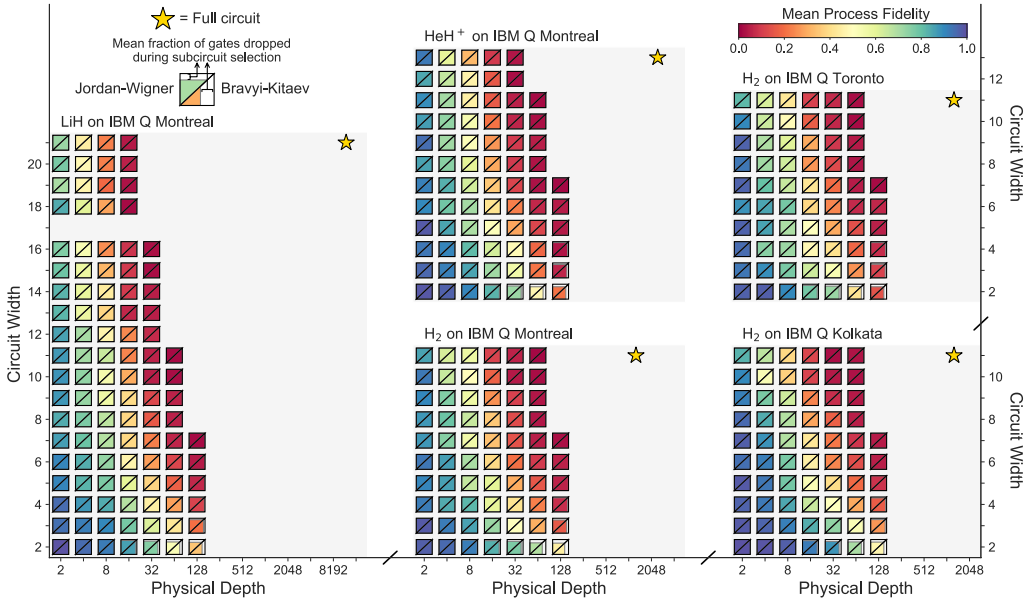


Proctor et al, Nat. Phys. 18, 75-79 (2022)

Algorithmic benchmarks

- Randomized benchmarks often won't predict other kinds of circuit.
- We need bespoke, application-specific benchmarks.
- There's a growing array of algorithmic benchmarking suites (e.g., the QED-C benchmarks[1]).
- Turning an algorithm into a benchmark is non-trivial:
  - Algorithms often have a lot of tunable "parameters", e.g., the problem instance.
  - Naïve benchmarks (just run the algorithms circuits) are often exponentially expensive to implement – solutions: mirror-circuit based methods[2] (for general circuits) and trap circuits[3] (for randomly compiled circuits).
  - Simple algorithmic benchmarks don't measure progress towards fault-tolerant implementations of algorithms.





[1]T. Lubinski et al., arXiv:2110.03137, [2]T. Proctor et al., arXiv:2204.07568, [3]S. Ferracin et al, PRA 104, 042603 (2021)

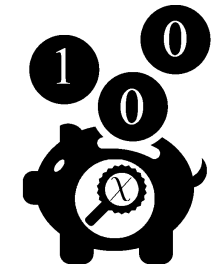
How do I run these benchmarks?

Quantum volume

Open source implementation in QisKit:

https://qiskit.org/textbook/ch-quantum-hardware/measuring-quantum-volume.html

Volumetric benchmarking & mirror circuit benchmarks



Open source implementation in pyGSTi with a QisKit interface:

https://github.com/pyGSTio/pyGSTi

pyGSTi repo contains Jupyter Notebook tutorials!

https://github.com/pyGSTio/pyGSTi/blob/master/jupyter_notebooks/Tutorials/algorithms/MirrorCircuitBenchmarks.ipynb

Algorithmic benchmarks

There are many different algorithmic benchmarking suites!

QED-C benchmarking suite: https://github.com/SRI-International/QC-App-Oriented-Benchmarks

You can create scalable algorithm benchmarks using mirror circuits via pyGSTi. See:

https://zenodo.org/record/6617686

Outline

1. Holistic benchmarking

- Volumetric benchmarking
- Algorithmic benchmarks

Many methods sit between these two extremes

- One/two-qubit RB
- Cycle benchmarking
- Pauli noise learning
- ….

2. Detailed error characterization

- State and process tomography
- Gate set tomography

High-level performance assessment
Aims to answer questions like:

- What's this device's failure rate when running many-qubit circuits?
- What algorithms can this device run?
- Is device X or device Y better?

Low-level performance assessment
Aims to answer questions like:

- What kinds of errors are occurring?
- What's the error rate of each kind of fundamental error on each gate?
- What's the process fidelity of this gate?

Standard tomography: state, process, and measurement tomography

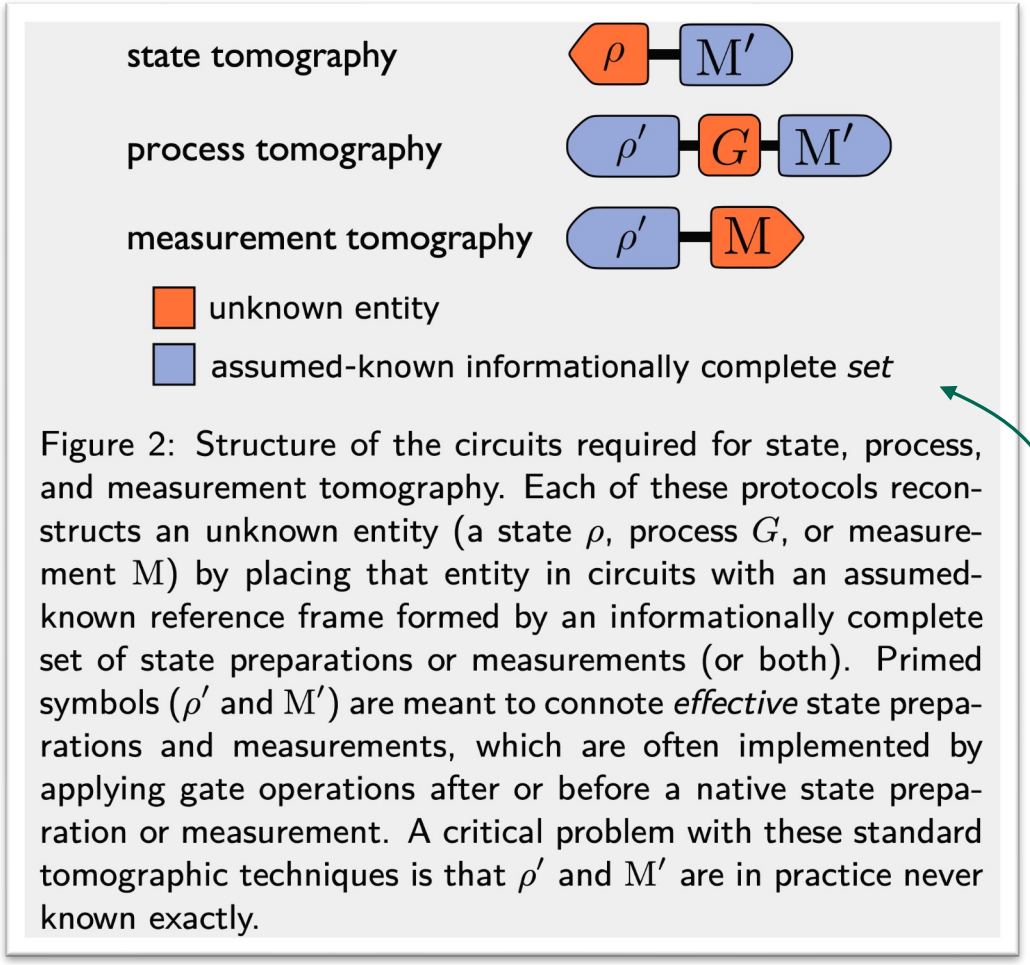


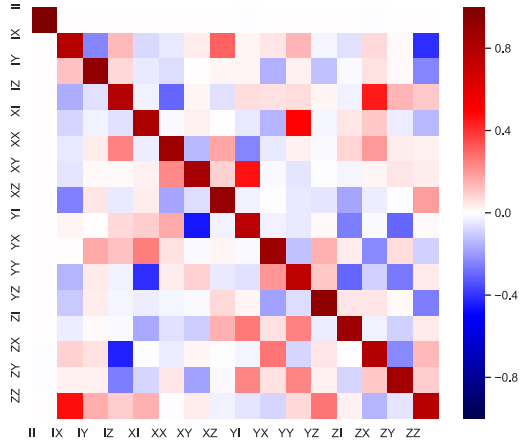
Figure 2: Structure of the circuits required for state, process, and measurement tomography. Each of these protocols reconstructs an unknown entity (a state ρ, process G, or measurement M) by placing that entity in circuits with an assumed-known reference frame formed by an informationally complete set of state preparations or measurements (or both). Primed symbols (ρ' and M') are meant to connote *effective* state preparations and measurements, which are often implemented by applying gate operations after or before a native state preparation or measurement. A critical problem with these standard tomographic techniques is that ρ' and M' are in practice never known exactly.

Nielsen et al, Quantum 5, 557 (2021)

- Tomography means learning all the elements of a vector (state tomography) or matrix (process tomography).

The normalized Pauli operators

$$E_{ij} = \mathrm{Tr}(P_j E[P_i])$$



- This just requires taking inner products with a known basis, and then applying a matrix inverse.
- This means using *informationally complete* (IC) measurements for state tomography, and IC state preparations and measurements for process tomography.

But there's a critical problem: pre-calibrated reference frames don't really exist!

Gate set tomography – tomography without pre-calibration

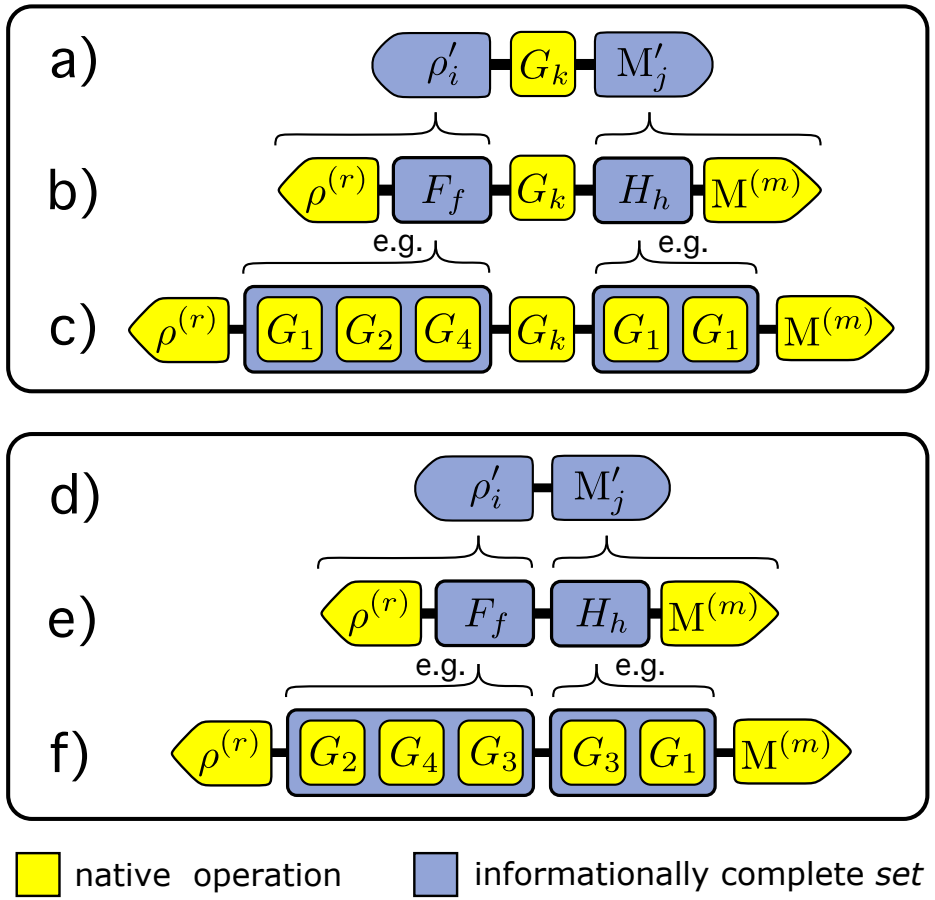
GST estimates all the elements of a *gate set*:

$$\mathcal{G} = \left\{ \left\{ |\rho^{(i)}\rangle\rangle \right\}_{i=1}^{N_\rho} ; \{G_i\}_{i=1}^{N_\mathrm{G}} ; \left\{ \langle\langle E_i^{(m)}| \right\}_{m=1,i=1}^{N_\mathrm{M}, N_\mathrm{E}^{(m)}} \right\}.$$

It does so up to a fundamental gauge freedom:

$$\begin{aligned} \langle\langle E_i^{(m)}| &\to \langle\langle E_i^{(m)}| M^{-1} \\ |\rho^{(i)}\rangle\rangle &\to M|\rho^{(i)}\rangle\rangle \\ G_i &\to M G_i M^{-1}. \end{aligned}$$

Linear gate set tomography



Process tomography experiments

Reference frame experiments

Nielsen et al, Quantum 5, 557 (2021)

Gate set tomography – tomography without pre-calibration

GST estimates all the elements of a *gate set*:

$$\mathcal{G} = \left\{ \left\{ |\rho^{(i)}\rangle\rangle \right\}_{i=1}^{N_\rho} ; \{G_i\}_{i=1}^{N_\mathrm{G}} ; \left\{ \langle\langle E_i^{(m)}| \right\}_{m=1,i=1}^{N_\mathrm{M},N_\mathrm{E}^{(m)}} \right\}.$$

It does so up to a fundamental gauge freedom:

$$\begin{aligned} \langle\langle E_i^{(m)}| &\to \langle\langle E_i^{(m)}| M^{-1} \\ |\rho^{(i)}\rangle\rangle &\to M|\rho^{(i)}\rangle\rangle \\ G_i &\to M G_i M^{-1}. \end{aligned}$$

Repeating "germs" enables Heisenberg-like estimation error!

Long-sequence gate set tomography

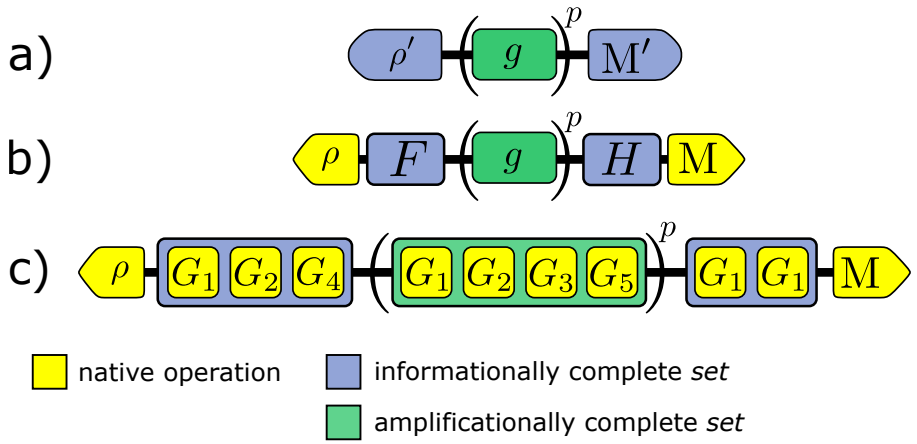


Figure 4: The structure of circuits in the standard GST experiment design, shown in increasing detail. **(a)** Each GST circuit consists of an effective state preparation ρ' (Eq. 52), followed by a *germ* circuit g repeated p times, followed by an effective measurement $\mathrm{M}' = \{E_i'\}$ (Eq. 51). **(b)** Effective preparations are often implemented by a native state preparation ρ followed by a *preparation fiducial circuit* F, and similarly effective measurements are often implemented by *measurement fiducial circuit* H followed by a native measurement M. **(c)** Writing the fiducials and germ in terms of native gate operations reveals how the native operations of a gate set compose to form a GST circuit.

Nielsen et al, Quantum 5, 557 (2021)

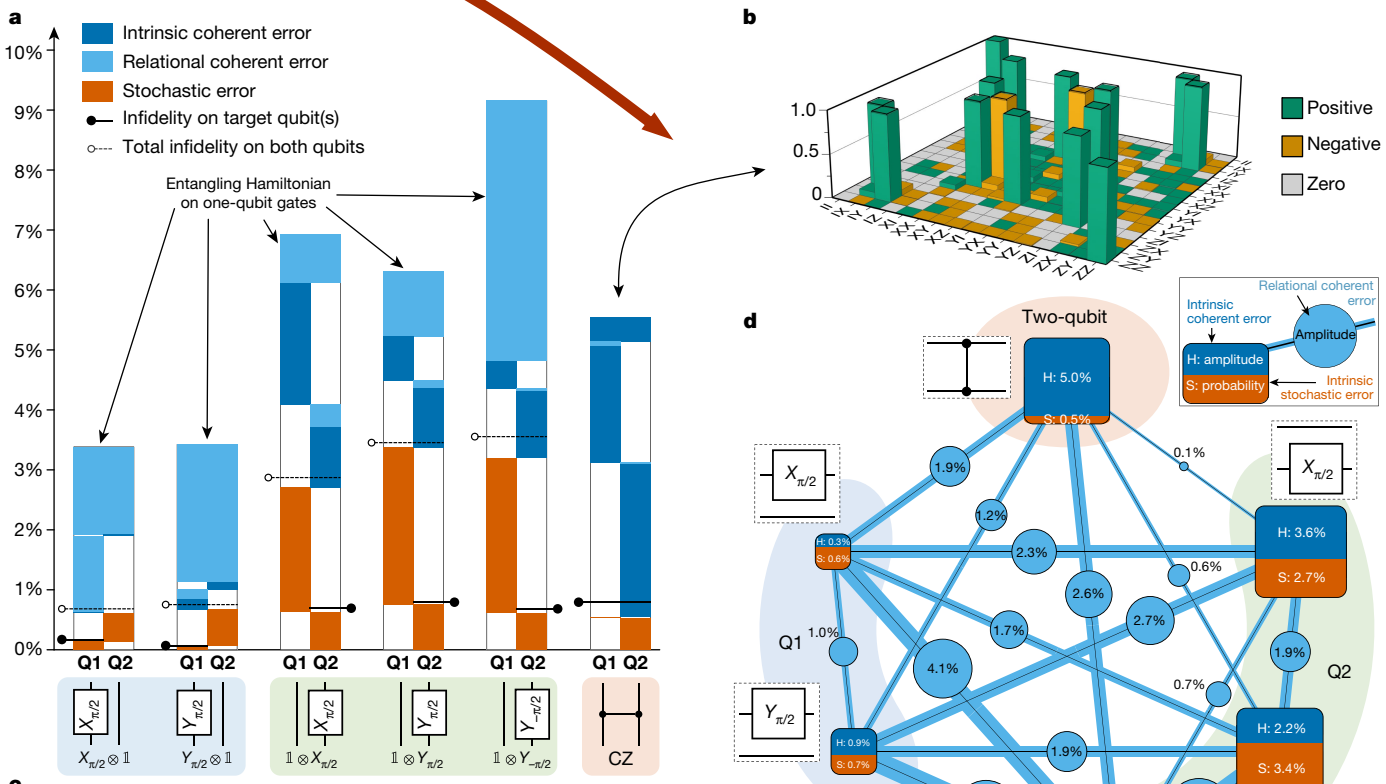
Process matrices are mysterious! How can we extract meaning from GST results?

Impenetrable process matrices can be turned into rates for each kind of a set of elementary error processes

Sector		Dimension	Action	Example effect (Bloch sphere)	ε_J (Error Probability)	θ_J (Error Amplitude)
Hamiltonian $\mathbb{H}$		d^2-1	$H_P[\rho] = -i[P,\rho]$		0	1
Stochastic (Pauli) $\mathbb{S}$		d^2-1	$S_P[\rho] = P\rho P - \mathbb{1}\rho\mathbb{1}$		1	0
Stochastic (Pauli-correlation) $\mathbb{C}$	{P,Q}=0	$\binom{d^2-1}{2}$	$C_{P,Q}[\rho] = P\rho Q + Q\rho P - \frac{1}{2}\{\{P,Q\},\rho\}$		0	0
	[P,Q]=0					1
Active $\mathbb{A}$	{P,Q}=0	$\binom{d^2-1}{2}$	$A_{P,Q}[\rho] = i\left(P\rho Q - Q\rho P + \frac{1}{2}\{[P,Q],\rho\}\right)$		0	1
	[P,Q]=0					0

Blume-Kohout et al., PRX Quantum 3, 020335 (2022)

This enables dividing the total error in a gate into different kinds of error – e.g., the coherent error on qubit 1.

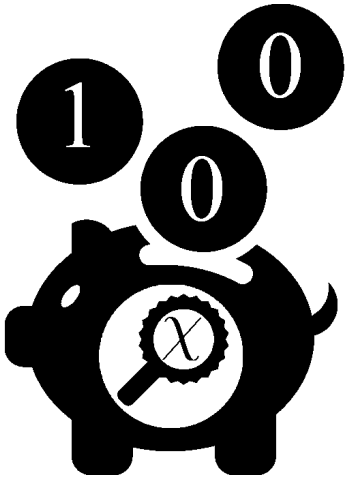


c

Gate	On-target average gate fidelity	Generator infidelity On Q1	On Q2	Total	Stochastic error	Intrinsic coherent error	Relational coherent error	Total error On Q1	On Q2	Total
$X_{\pi/2}\otimes\mathbb{1}$	**99.87(2)%**	**0.17(3)%**	0.47(5)%	0.68(6)%	0.60(6)%	0.34(14)%	2.8(3)%	2.0(3)%	0.71(15)%	3.4(3)%
$Y_{\pi/2}\otimes\mathbb{1}$	**99.95(2)%**	**0.07(3)%**	0.61(6)%	0.75(6)%	0.67(6)%	0.9(3)%	2.6(3)%	1.0(2)%	1.2(4)%	3.4(3)%
$\mathbb{1}\otimes X_{\pi/2}$	**99.48(3)%**	2.14(5)%	**0.69(4)%**	2.86(7)%	2.69(6)%	3.6(3)%	2.2(3)%	5.0(3)%	3.05(17)%	6.9(3)%
$\mathbb{1}\otimes Y_{\pi/2}$	**99.41(5)%**	2.63(8)%	**0.79(6)%**	3.44(10)%	3.36(10)%	2.3(4)%	1.9(4)%	4.1(6)%	2.6(2)%	6.3(5)%
$\mathbb{1}\otimes Y_{-\pi/2}$	**99.49(5)%**	2.60(13)%	**0.68(7)%**	3.54(16)%	3.18(15)%	3.1(4)%	5.1(5)%	4.2(7)%	3.2(3)%	9.1(5)%
CZ	**99.37(11)%**	0.11(10)%	0.66(11)%	**0.79(14)%**	0.54(13)%	4.9(3)%	0.7(3)%	3.2(4)%	4.1(3)%	5.5(4)%

M. Madzik et al., Nature 601, 348 (2022)

How do I run GST?



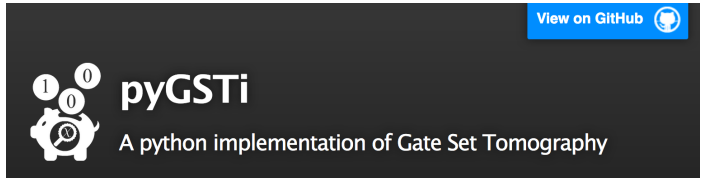
One- and two-qubit GST

Open source implementation in pyGSTi:

https://github.com/pyGSTio/pyGSTi

pyGSTi repo contains Jupyter Notebook tutorials!

https://github.com/pyGSTio/pyGSTi/blob/master/jupyter_notebooks/Tutorials/algorithms/GST-Overview.ipynb



Many-qubit GST

GST on many qubits is an in-development technique!

Thanks!

https://qpl.sandia.gov/

At the QPL we spend our time developing, understanding and using QCVV+ techniques – from benchmarks to GST and beyond – and we collaborate with experimentalists across the world to understand cutting-edge hardware.

The QPL is hiring postdocs!

So if you're excited to work on QCVV+ research now or in the future please get in touch.

tjproct@sandia.gov

This work was funded in part by Sandia's Laboratory Directed Research and Development program. This material was funded in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research Quantum Testbed Program. All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of the U.S. Government. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.