

# Not All Benchmarks Are Created Equal

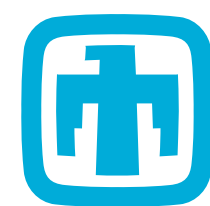
April 3, 2020

Robin Blume-Kohout

Erik Nielsen, Timothy Proctor, Kenny Rudinger, Kevin Young, Mohan Sarovar



U.S. DEPARTMENT OF  
**ENERGY**



Sandia National Laboratories

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-NA-0003525



# Abstract

Testbed-class quantum computers -- fully programmable 5-50 qubit systems -- have burst onto the scene in the past few years. The associated surge in funding, hype, and commercial activity has spurred interest in “benchmarks” for assessing their performance.

Unsurprisingly, this has generated both a number of scientifically interesting ideas *and* a lot of confusion and kerfuffle. I will try to explain the state of play in this field -- known historically as “quantum characterization, verification, and validation (QCVV)” and more recently and generally as “quantum performance assessment” -- by:

- briefly reviewing its history,
- explaining the different categories of benchmarks and characterization protocols, and
- identifying what they're good for.

The overarching message of my talk is that these are distinct tools in a diverse toolbox -- almost every known protocol and benchmark really measures a distinct and particular thing, and we probably need *more* of them, not fewer.

# I made a Bibliography

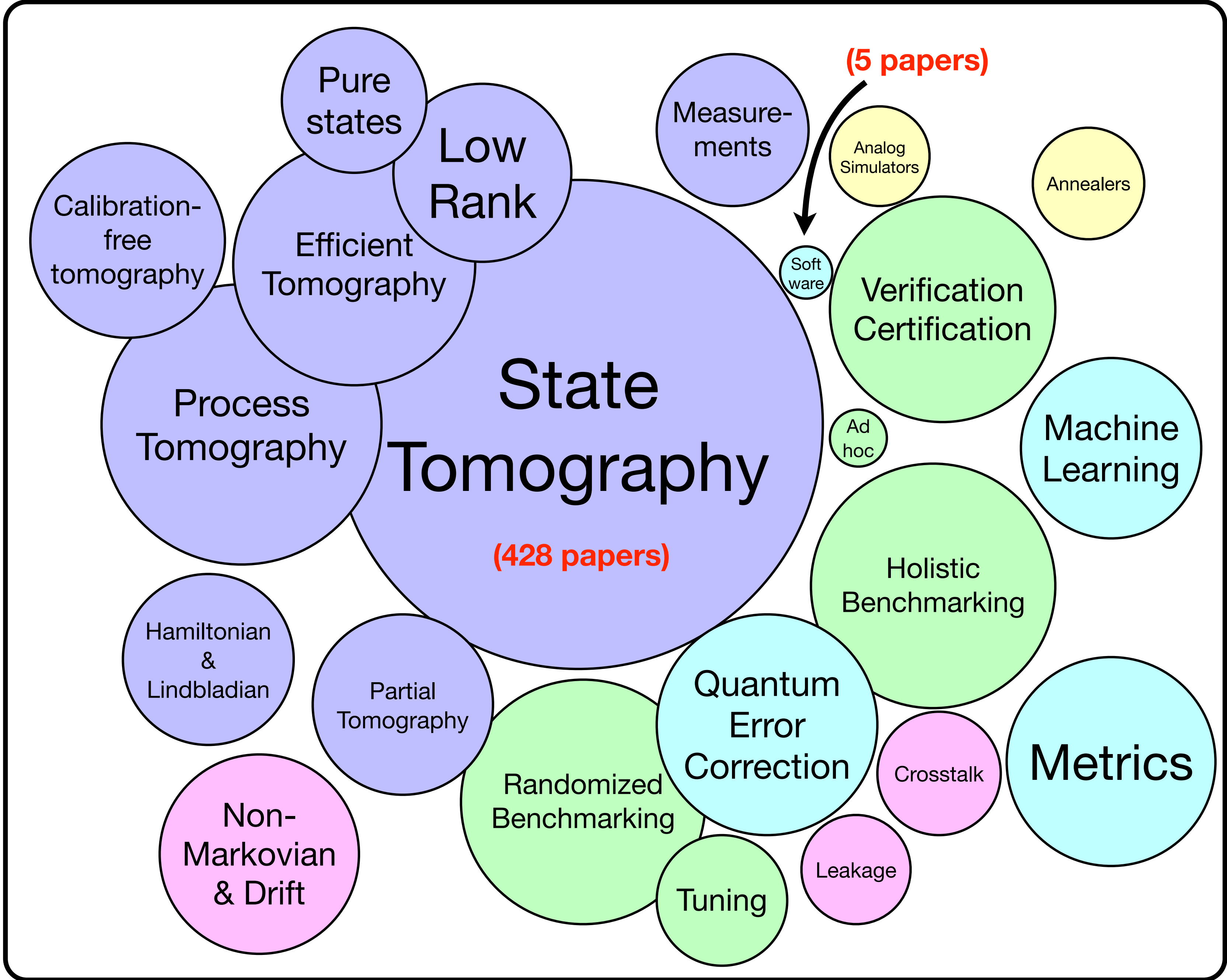
Early this year, I set out to collect *every* paper every written about *assessing performance of digital quantum computing components*, which is my best attempt to describe my scientific briar patch.

As of March 18, I collected 1535 sources from 1957 - 2020.

- 1170 journal articles
- 276 arxiv preprints
- 37 PhD and masters theses
- 29 conference papers
- 13 book chapters
- 7 books
- 1 NASA report, 1 set of lecture notes, and 1 poster.

# I classified them.

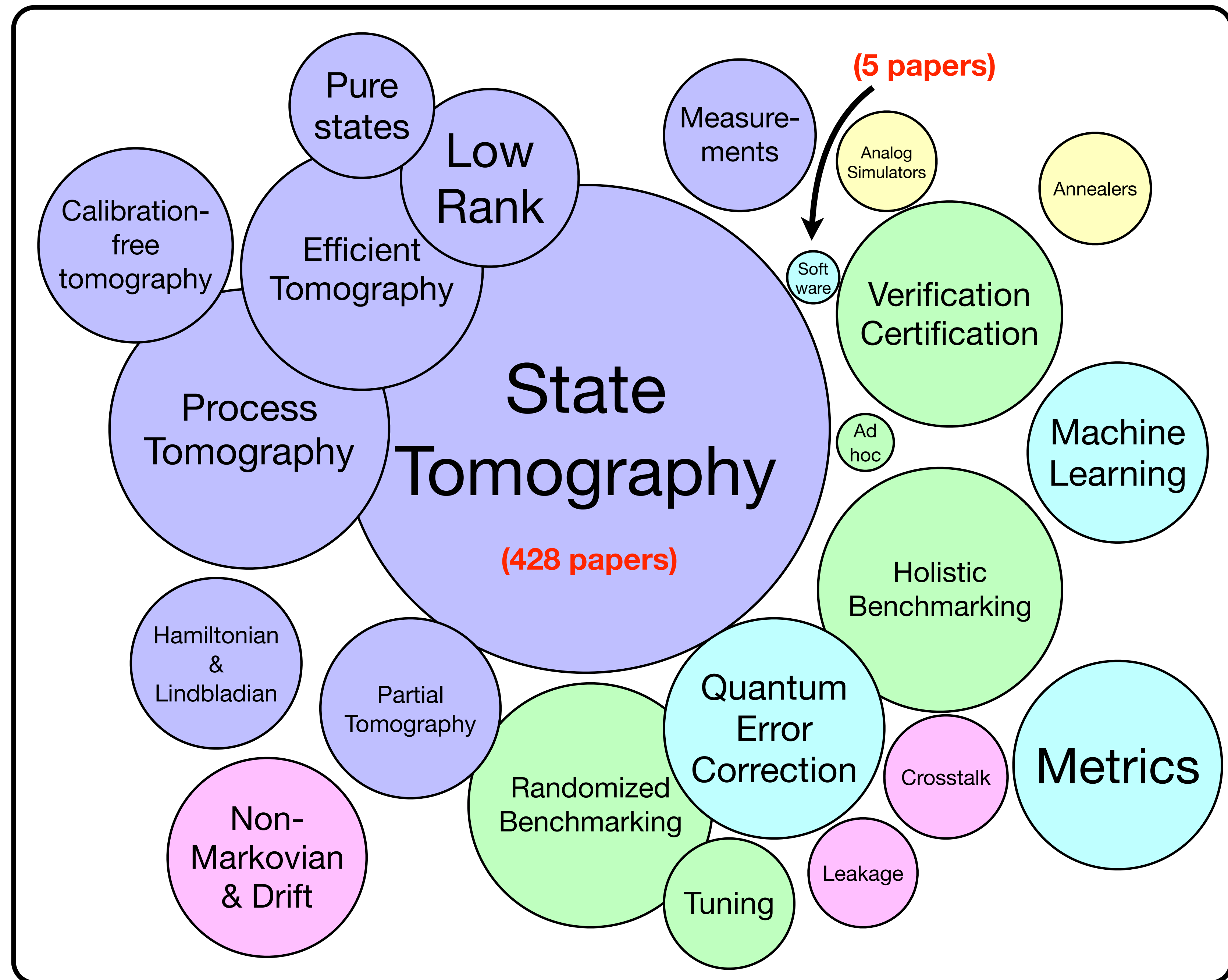
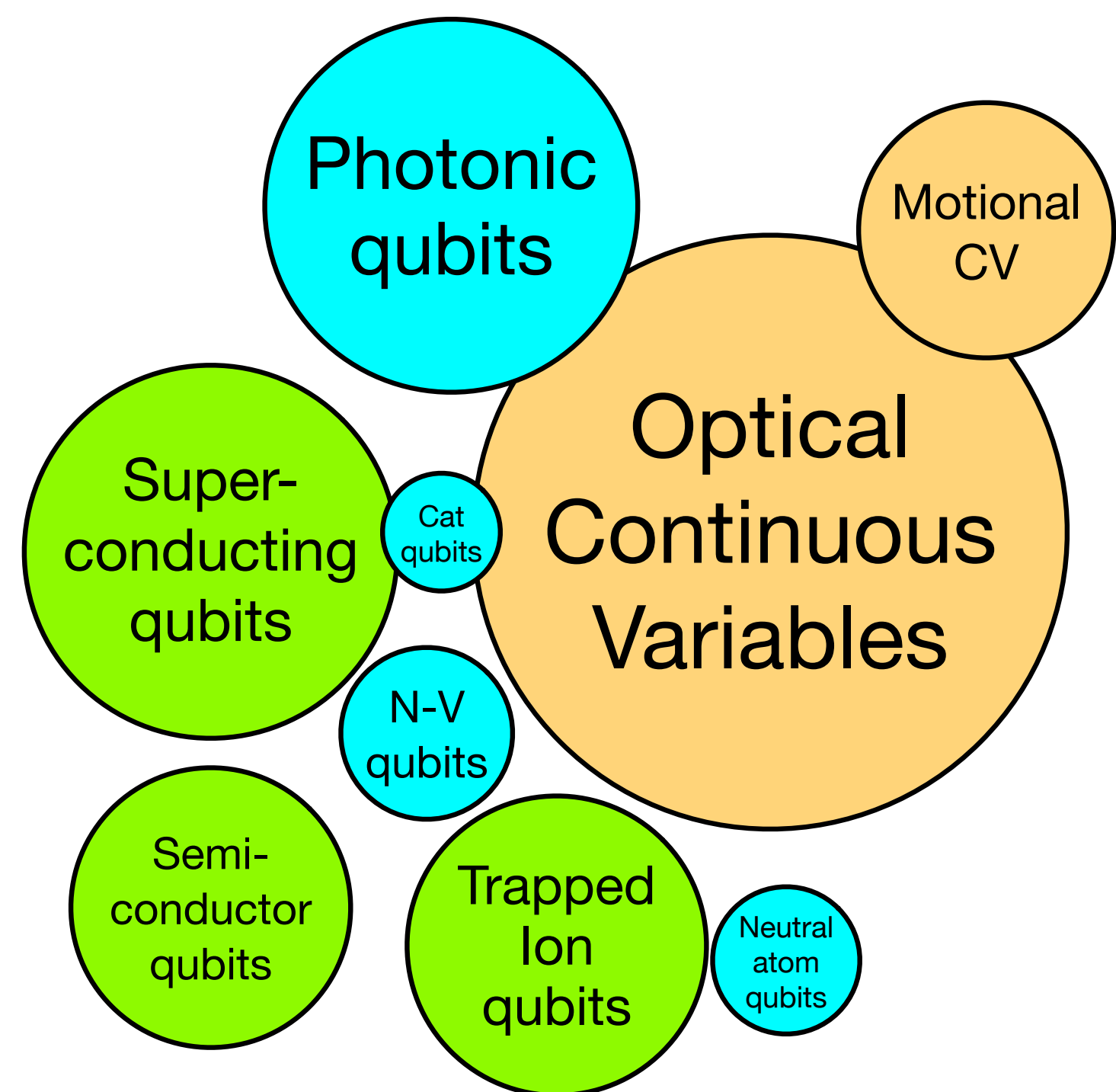
The number of papers in a category is proportional to the *area* of the circle.



# I classified them.

The number of papers in a category is proportional to the *area* of the circle.

I also classified experimental papers by system type.



# Does this define what's important?

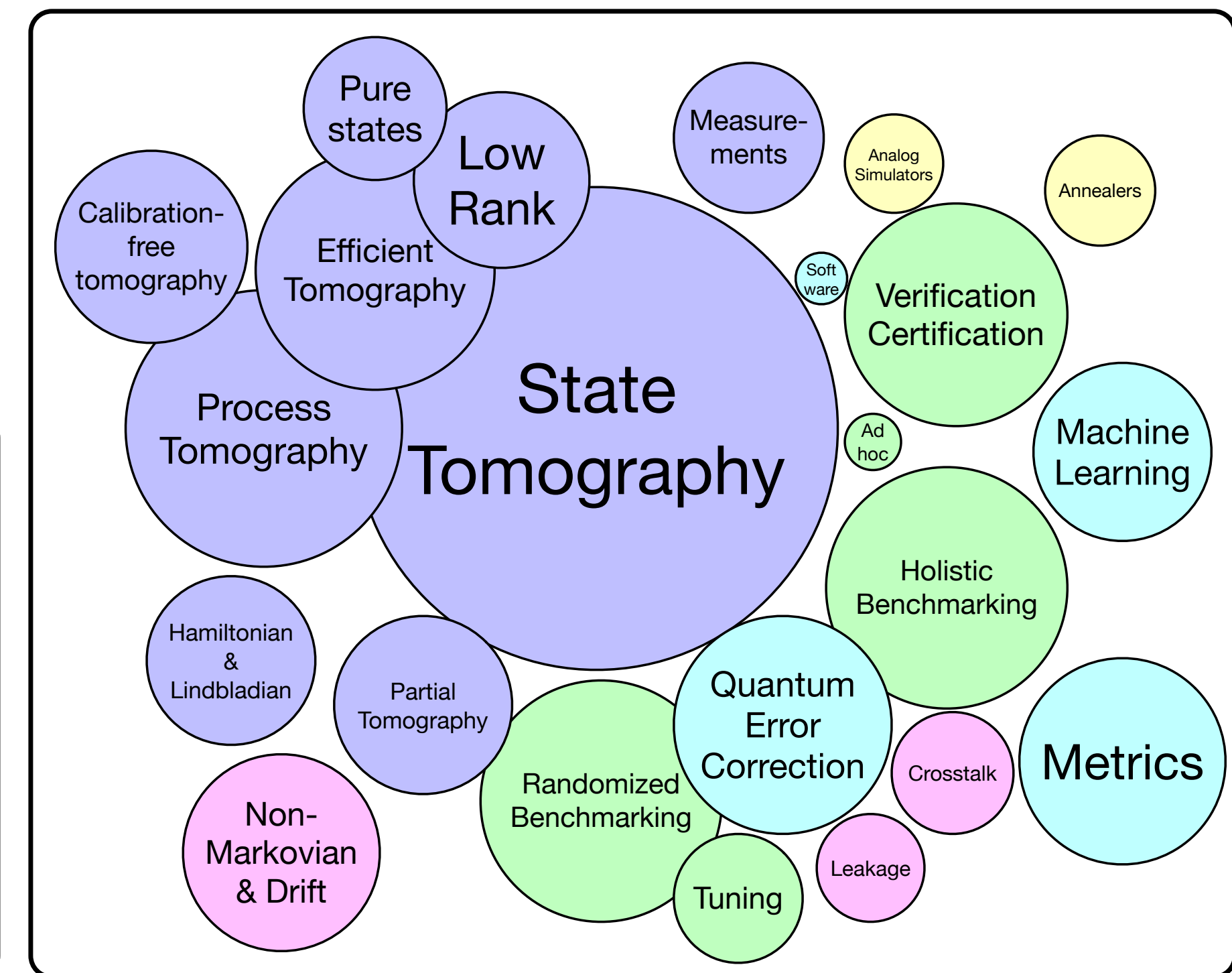
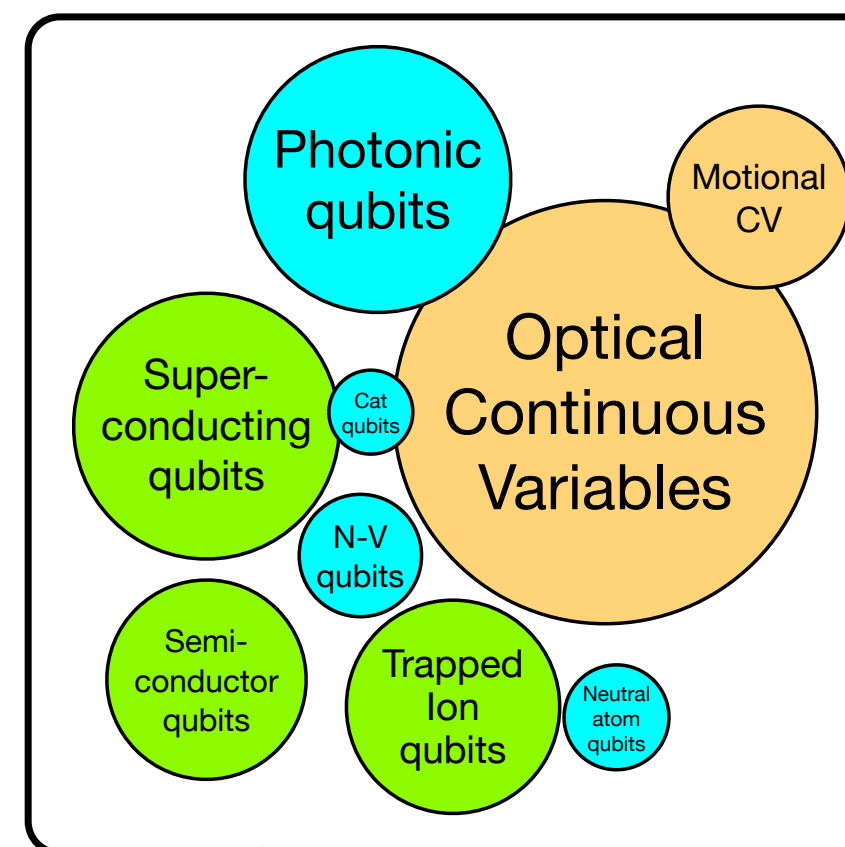
Taken at face value, this seems to suggest that we should spend the next hour talking about state tomography and optical CV systems.

But this is just a historical artifact.

Scientists spent 20 years writing about tomography of Wigner functions because they didn't have anything better to do.

But now we have *actual* quantum computers (tiny ones with 5-50 qubits).

What's important *now*?



# ‘Metrics & Benchmarks’ are very exciting

Quantum computing has gained a lot of funding, hype, and engineering.

Now many people want metrics and benchmarks, in order to:

- Quantify how good quantum computers are,
- Determine which of two quantum computers is better,
- Measure progress (toward betterness) over time,
- ???
- Profit!

Maybe these are not the best questions to ask.

Let’s consider some questions that “benchmarks” could address.

# Some questions we'd like answered

Does this thing work?

Does this thing do  
what you said it does?

Is this thing good enough?

How well does this thing work?

Which of these things is better?

What does this thing do?

How can I make this thing better?

What's wrong with this thing?

What could I do with this thing?

What will happen when I use this thing?

How many of these things would I need?

What *can* this thing do?



# Some questions we'd like answered

## Testing

Does this thing work?

Does this thing do  
what you said it does?

Is this thing good enough?

How well does this thing work?

Which of these things is better?

What does this thing do?

How can I make this thing better?

What's wrong with this thing?

What could I do with this thing?

What will happen when I use this thing?

How many of these things would I need?

What *can* this thing do?

# Some questions we'd like answered

## Testing

Does this thing work?

Does this thing do  
what you said it does?

Is this thing good enough?

How well does this thing work?

Which of these things is better?

What does this thing do?

How can I make this thing better?

What's wrong with this thing?

What could I do with this thing?

What will happen when I use this thing?

How many of these things would I need?

What *can* this thing do?

## Benchmarking

# Some questions we'd like answered

## Testing

Does this thing work?

Does this thing do  
what you said it does?

Is this thing good enough?

How well does this thing work?

Which of these things is better?

## Characterization

What does this thing do?

How can I make this thing better?

What's wrong with this thing?

What could I do with this thing?

What will happen when I use this thing?

How many of these things would I need?

What *can* this thing do?

## Benchmarking

# Protocols est omnis divisa in partes tres

## Tests

- Certify or verify a property
- Report “yes” or “no”.
- Are conceptually simplest
- Can be fraught in practice

## Characterization Protocols

- Measure all aspects of a thing’s function
- Provide rich detail about faults/behavior
- Intended to *predict* all possible uses
- Generally require the most effort & data

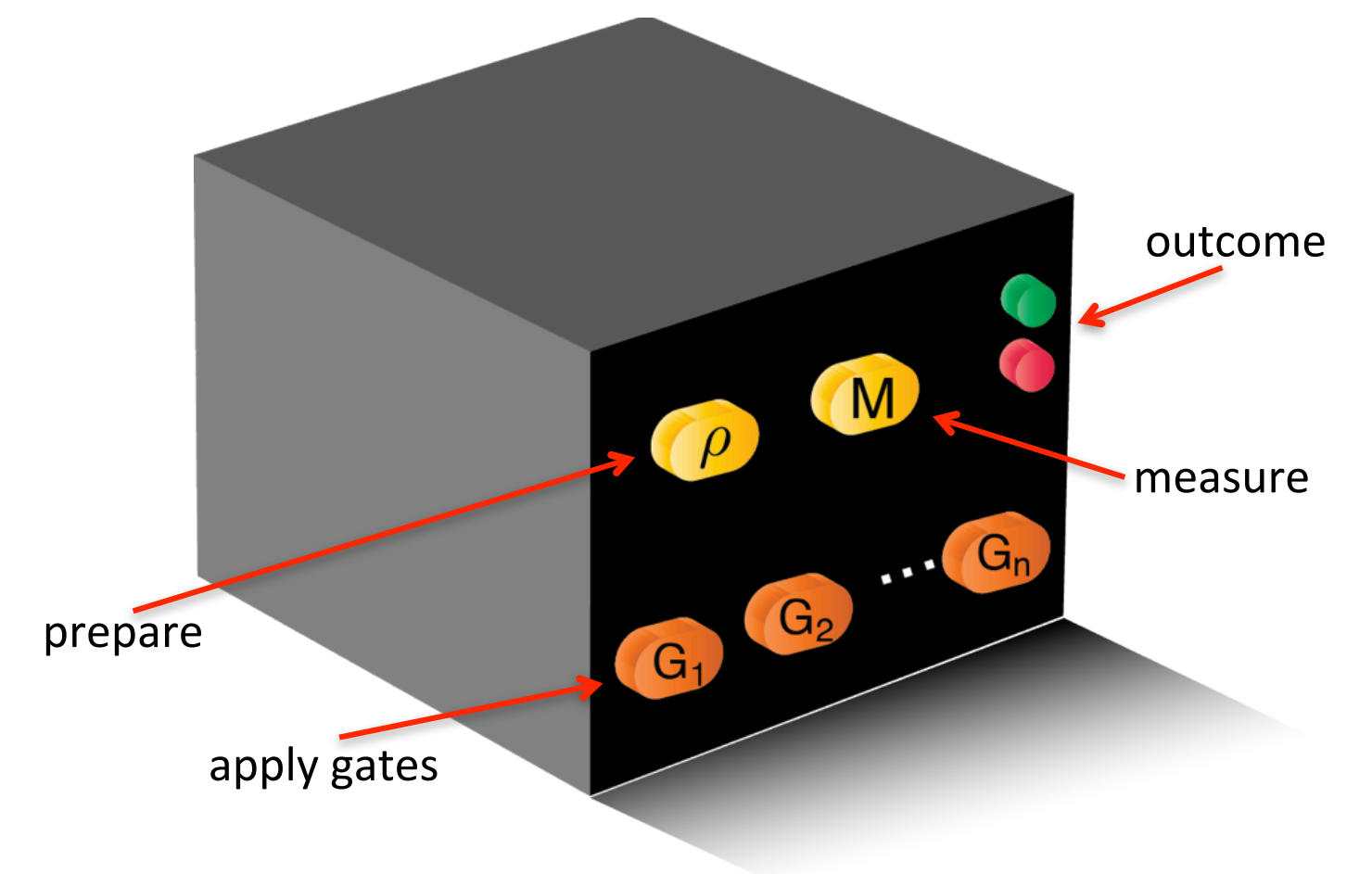
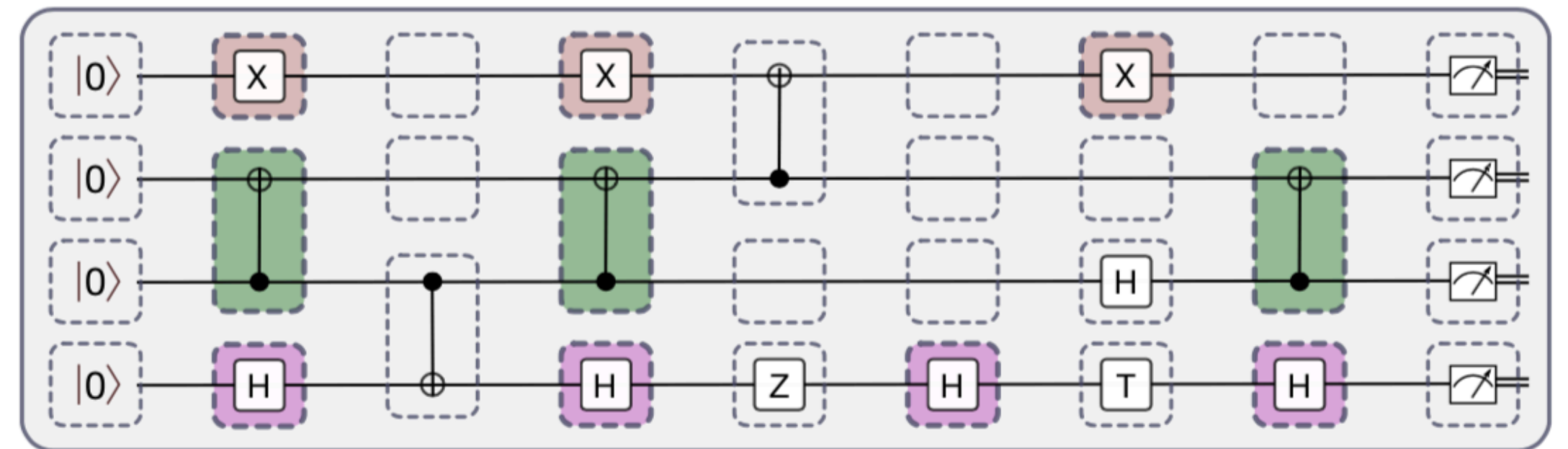
## Benchmarks

- Quantify how good something is, or how well it performs its function.
- Report one or more quantitative numbers.
- Are well-suited for comparing different things that do the same task.
- Are usually *task-specific* and don’t generally predict other tasks.

# How $\{T / B / C\}$ protocols work

Broadly speaking, all these protocols follow the same pattern:

1. Run a test suite of circuits.
2. Repeat each circuit many times and record statistics
3. Analyze the results.
4. Estimate vectors/matrices/scalars or perform hypothesis testing.
5. Draw some conclusion about error performance.



In contrast to much CS literature, we are usually not worried about cryptographic security or “Mechanical Turk” problems — we assume the data is honest and faithful. But we do worry about systematic errors!

# What “things” are we talking about?

*Does this thing work?*

*What does this thing do?*

*How well does this thing work?*

Quantum computers, obviously. Little ones now, bigger ones later.

But also *components* of quantum computers.

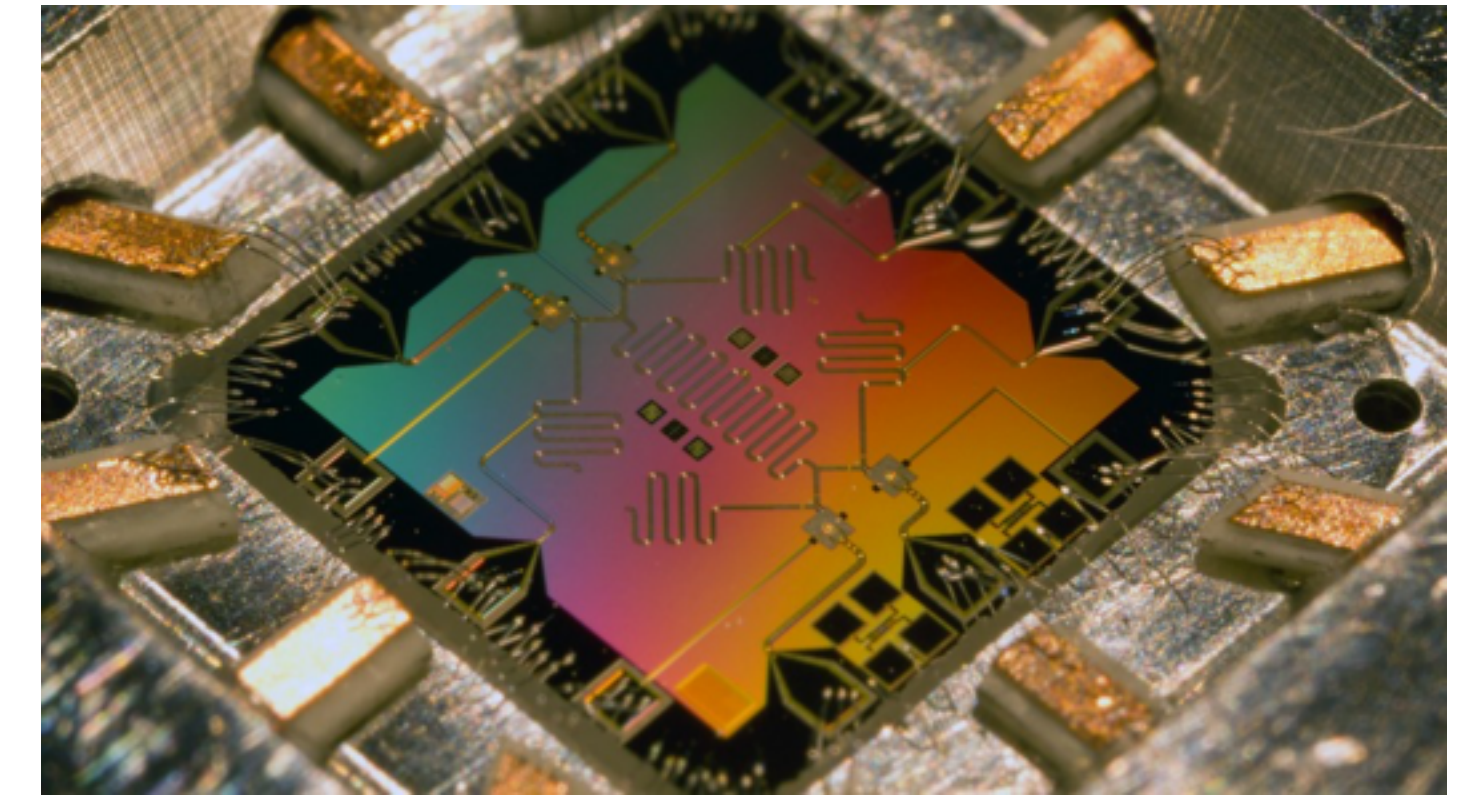
- qubits, subsets of qubits, or *qudits*
- individual logic operations
- important subroutines
- and (in a slightly different sense), entire algorithms

And also (throughout the literature) *other* things with no relation to quantum computing, like cavity modes and communication channels.

# Examples of “fully integrated” devices

Testbed-class digital quantum computers

- 5-50 qubits (2?)
- Integrated programmable systems:  
qubits + control + environment
- Designed to run general circuits

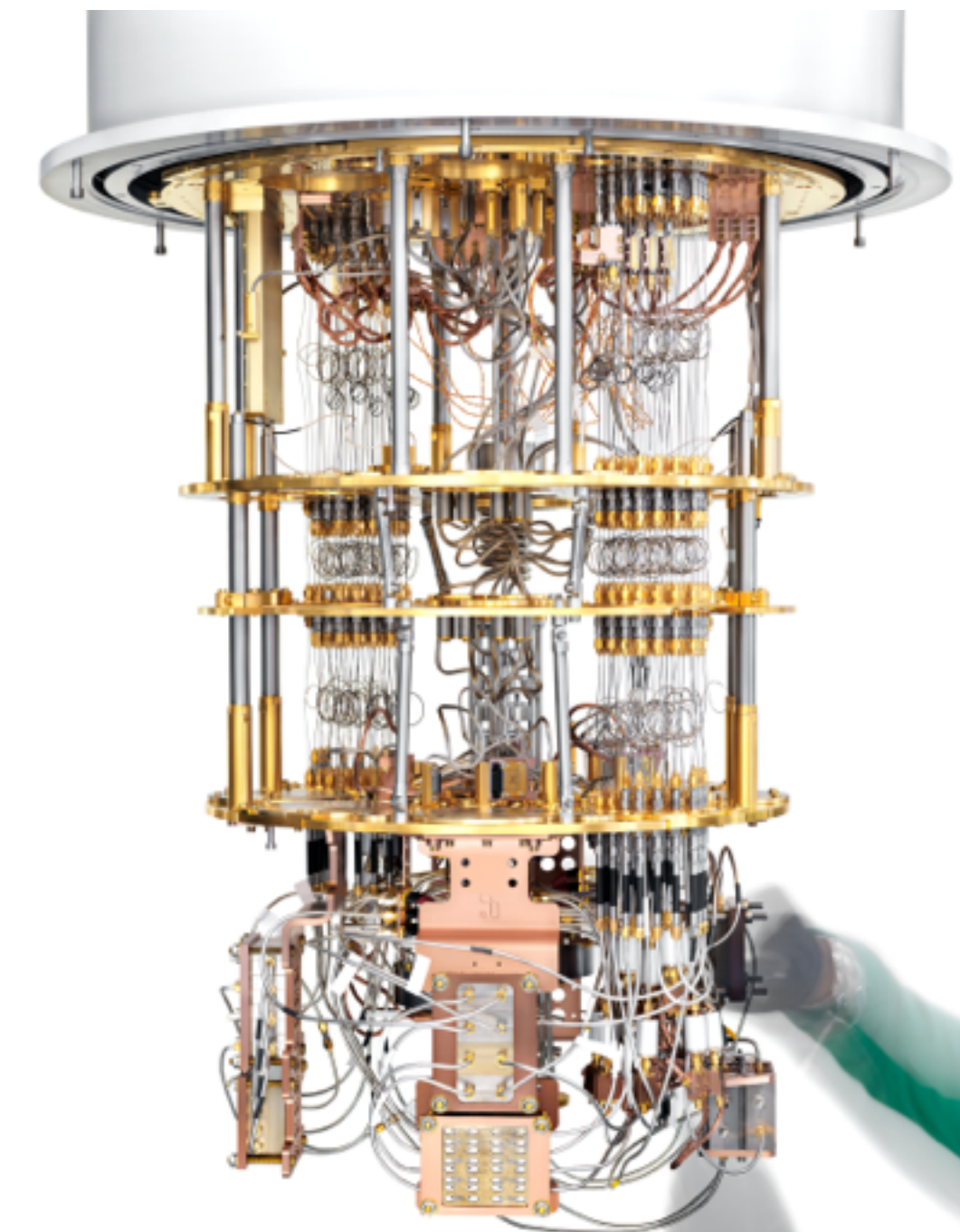
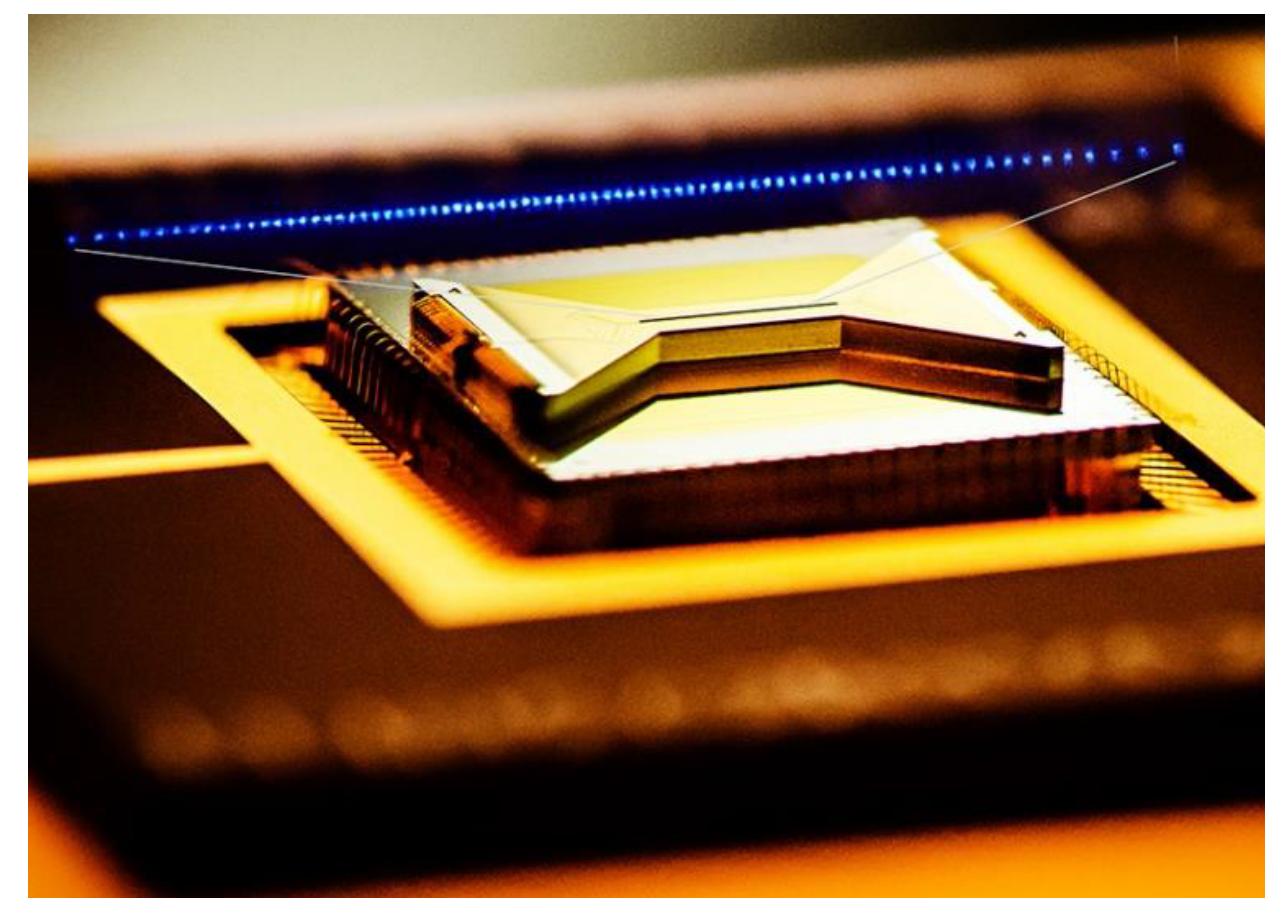


Future digital quantum computers

- NISQ platforms (50+ qubits)
- Extrapolation into the future...

Analogue simulators

Quantum annealers (D-Wave)



# Examples of “fully integrated” devices

Testbed-class digital quantum computers

- 5-50 qubits (2?)
- integrated programmable systems:  
qubits + control + environment
- designed to run general circuits

Future digital quantum computers

- NISQ platforms (50+ qubits)
- extrapolation into the future...

Analogue simulators

Quantum annealers (D-Wave)

**There are almost *no* good reasons to compare these devices to each other.**



# We test, benchmark, characterize at 3 scales

## **Integrated Quantum Processors**

Small (2-10)

Medium (10-50)

Large (50+)

## **Composite Components**

Gate set (1-2 qubits)

Subroutine

Parity check

(stabilizer meas.)

QEC cycle (LogiQ)

## **Elementary Operations**

1-qubit gates










2-qubit gates

Initialization/reset

Measurements

Non-computational  
operations (shuttle,  
hide, cool, etc).

# Which methods for which components?

	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing	 what would you test?	 usually need more detail	not useful  never perfect
Benchmarking			 usually need more detail
Characterization	 too big	 feasible? necessary?	

# Characterizing > Benchmarking > Testing

Benchmark metrics can usually be computed from full characterization.

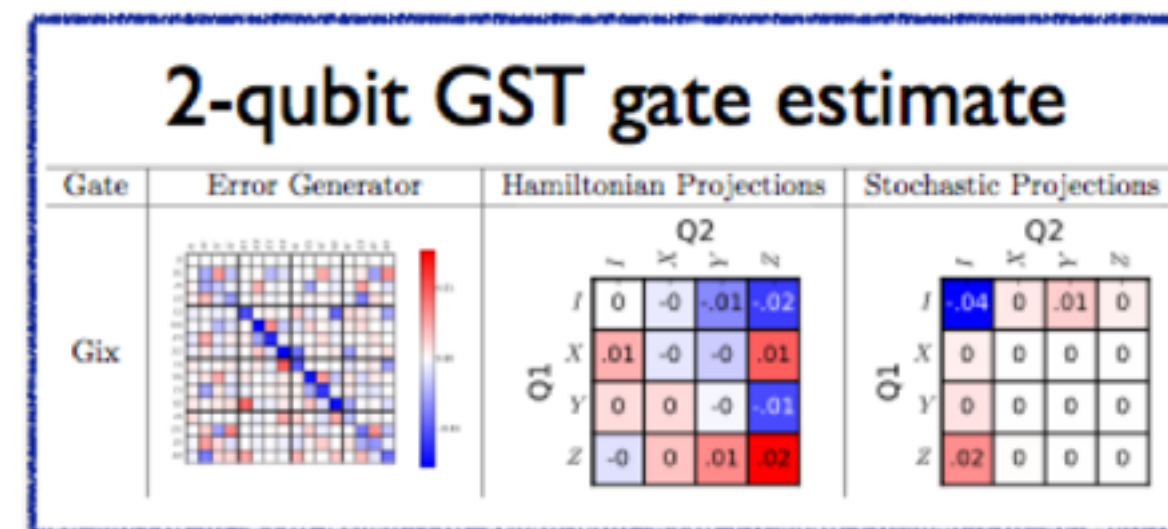
Binary property tests can usually be inferred from metrics.

This is usually wasteful and expensive:

- More detailed characterization requires measuring more things.
- Statistical uncertainty increases with more detail.

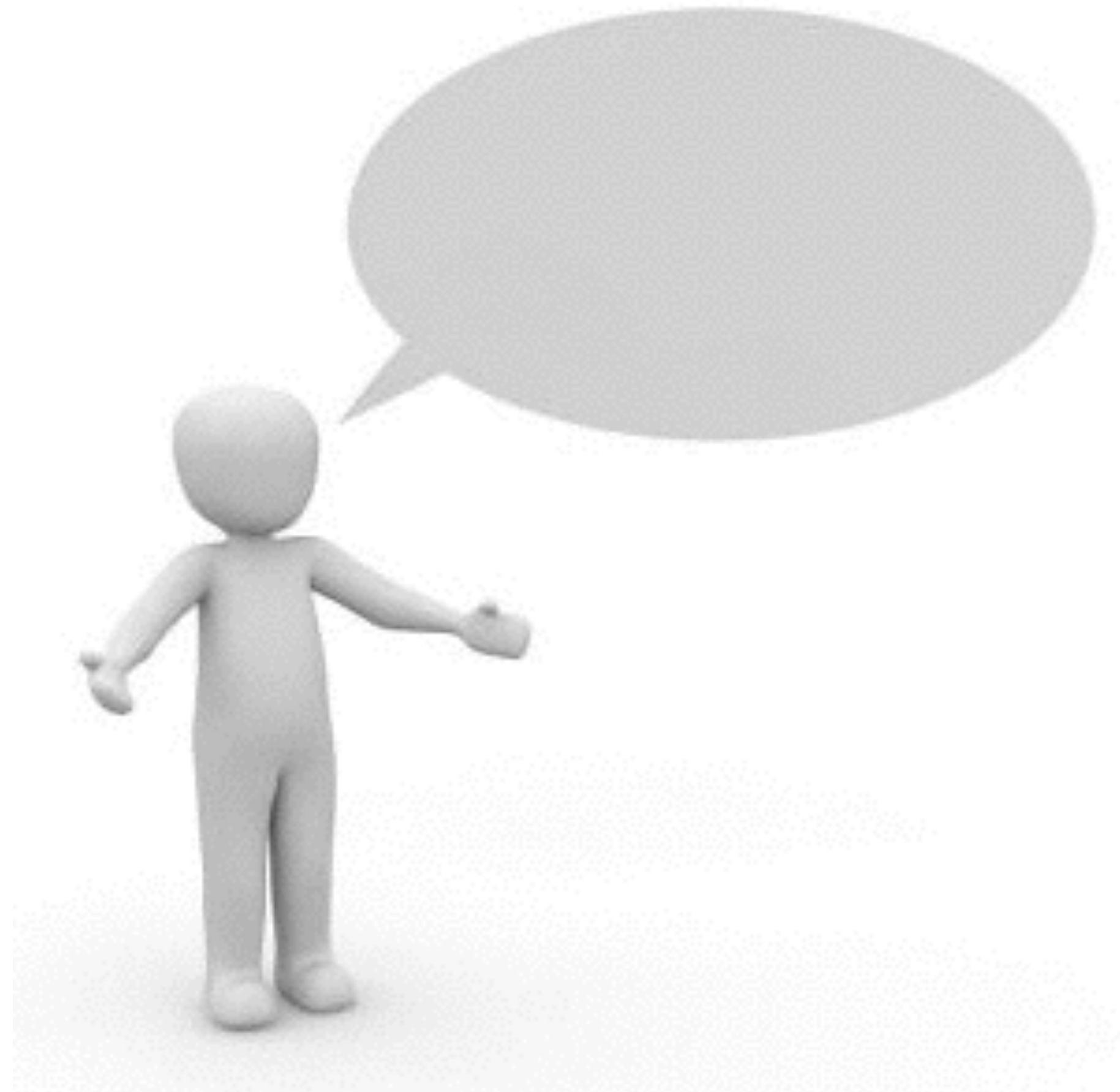
But some metrics/properties are only known as complicated functions of a detailed characterization, e.g.:

- entanglement
- diamond norm
- logical error rate



→  $F \approx 0.99$  → **good!**  
→ **bad**

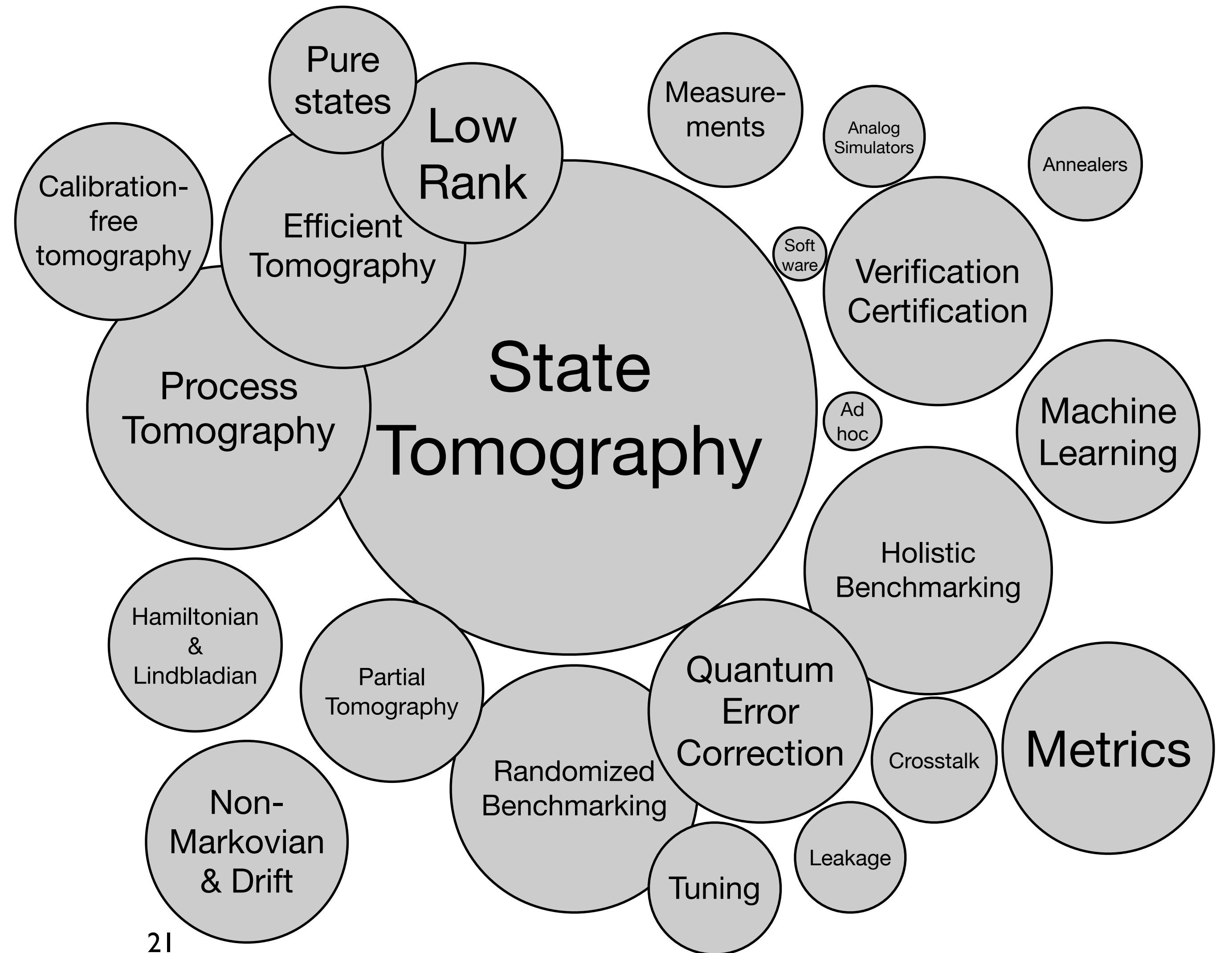
**What does it mean for an  
operation, component, or processor to “work”?**



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

# Focus:

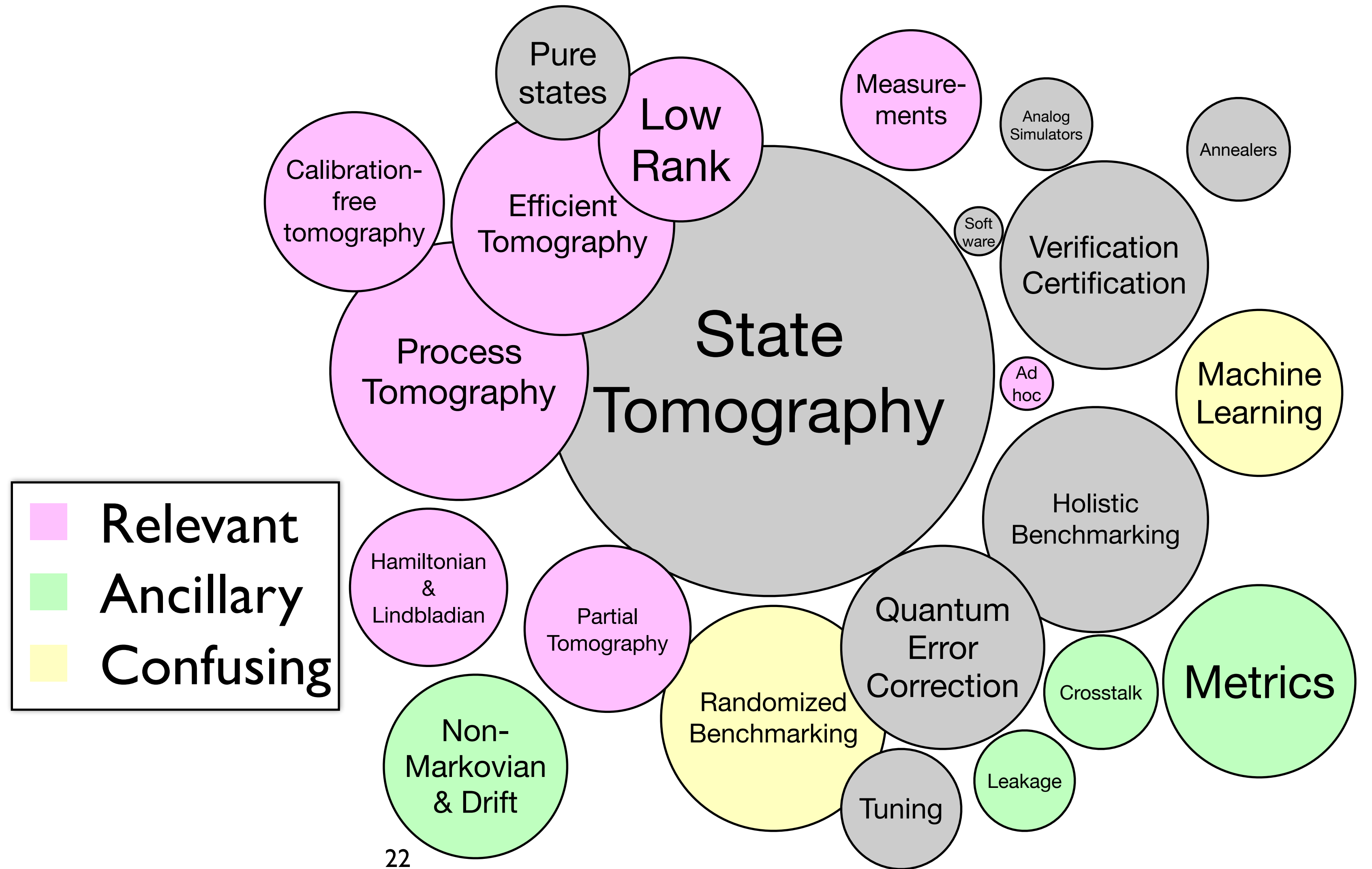
## *Characterizing Operations*



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

# Focus:

## *Characterizing Operations*



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

# Focus:

## *Characterizing Operations*

Process tomography

Detector tomography

Gate set tomography

Compressed sensing

Hamiltonian estimation

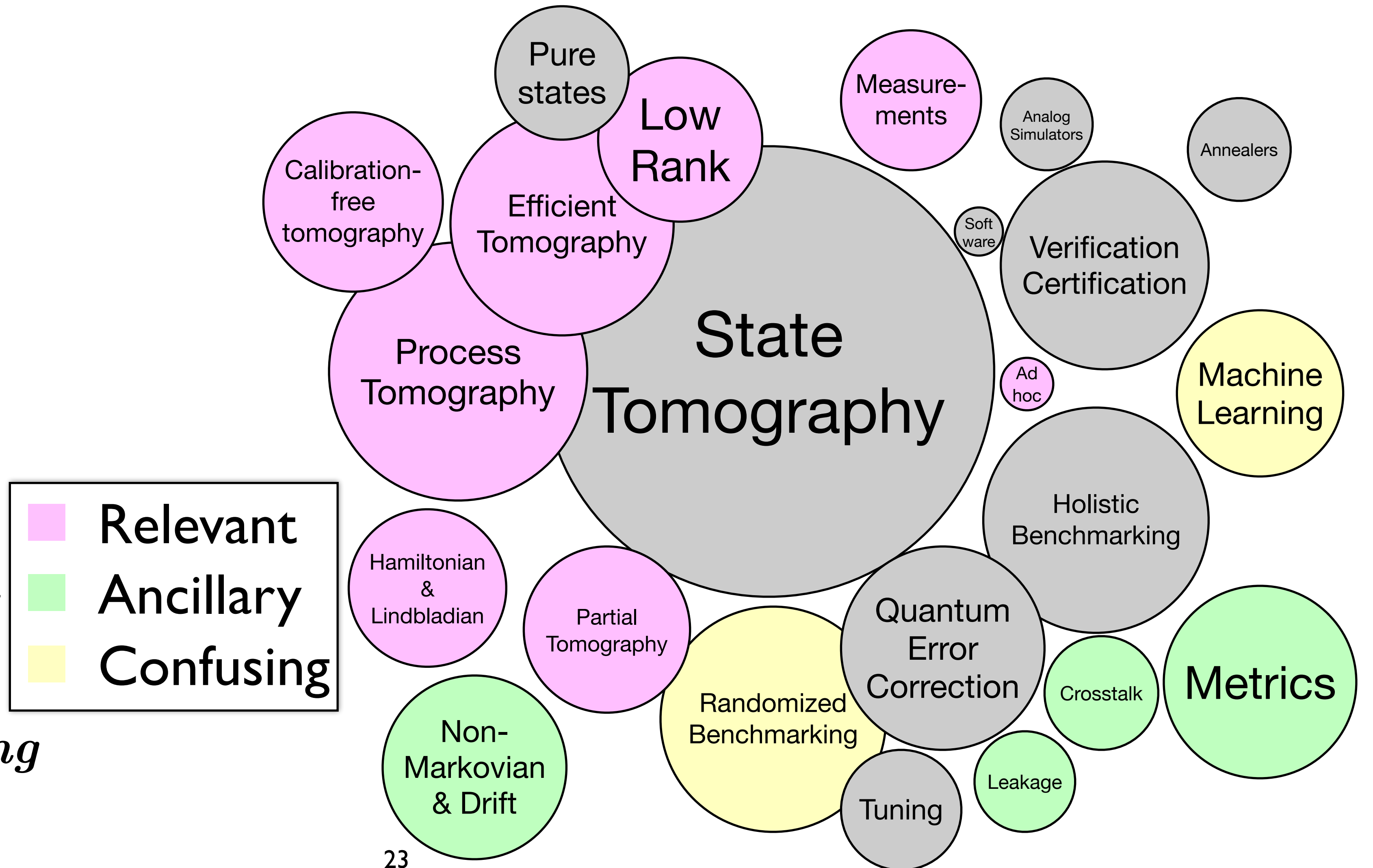
Robust phase estimation

Direct fidelity estimation

[Ad hoc methods]

*Randomized benchmarking*

*Machine learning*



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

# Focus:

## *Characterizing Operations*

*Note:* There is significant overlap and ambiguity in experiments and the literature between *characterizing* operations and *benchmarking* operations.

Tomography is often reduced to a benchmarking procedure, by ignoring the detailed results and just extracting a fidelity.

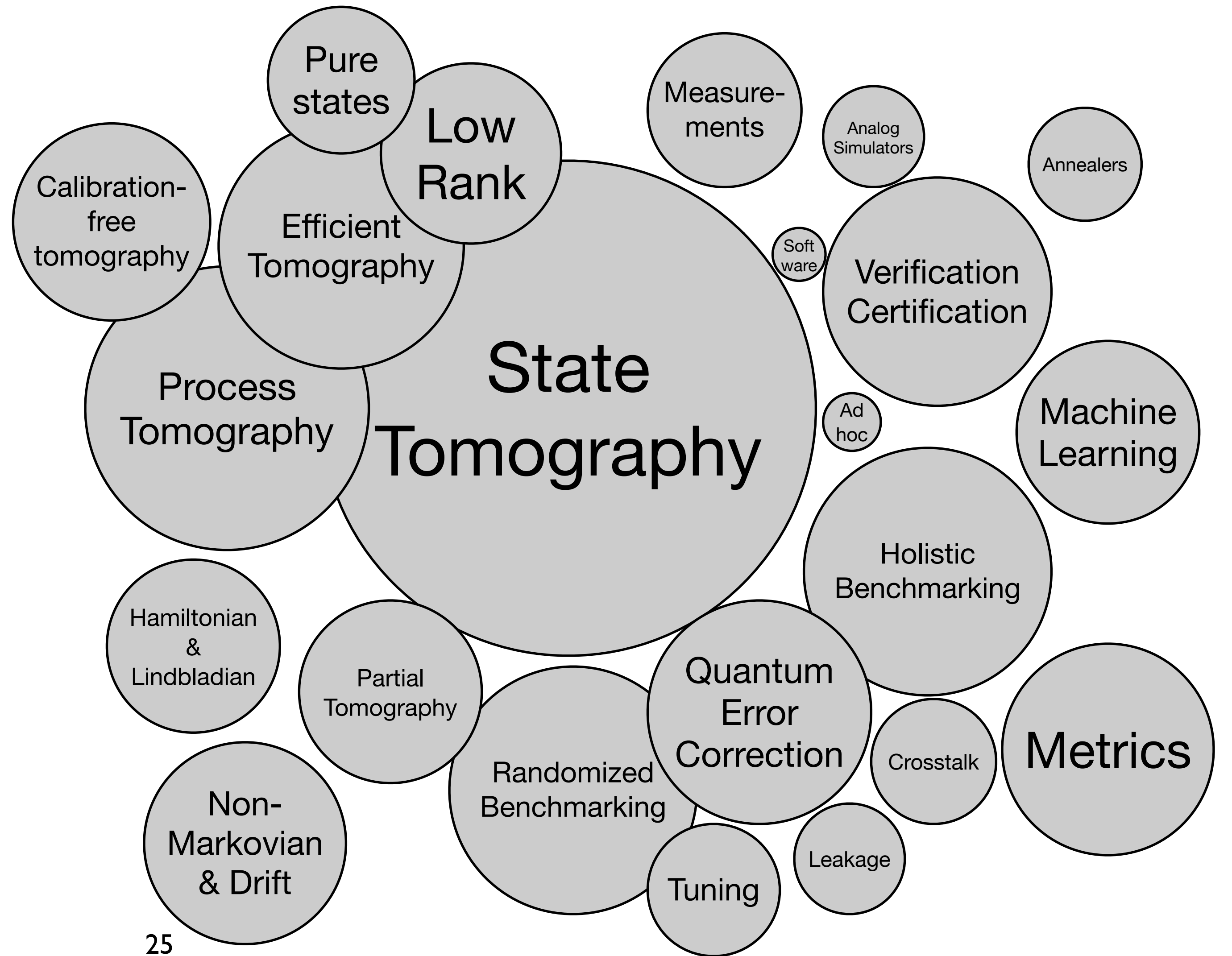
Randomized benchmarking is often promoted to a *de facto* characterization procedure by using RB error rates in predictive models.



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

**Focus:**

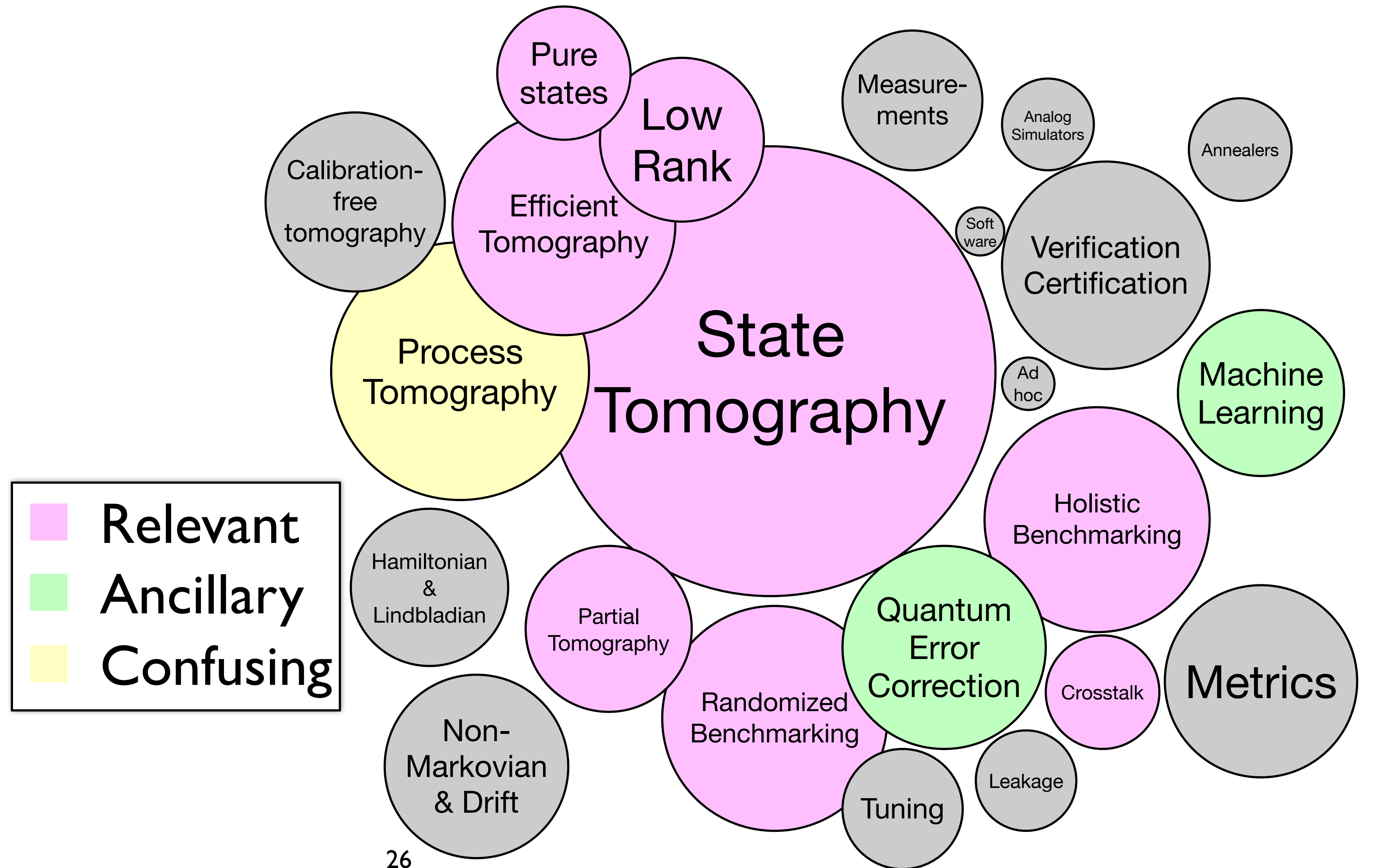
# *Benchmarking Subroutines*



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

# Focus:

## *Benchmarking Subroutines*



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

**Focus:**

# *Benchmarking Subroutines*

**State tomography (!)**

- compressed sensing
- MPS tomography
- pure state tomography

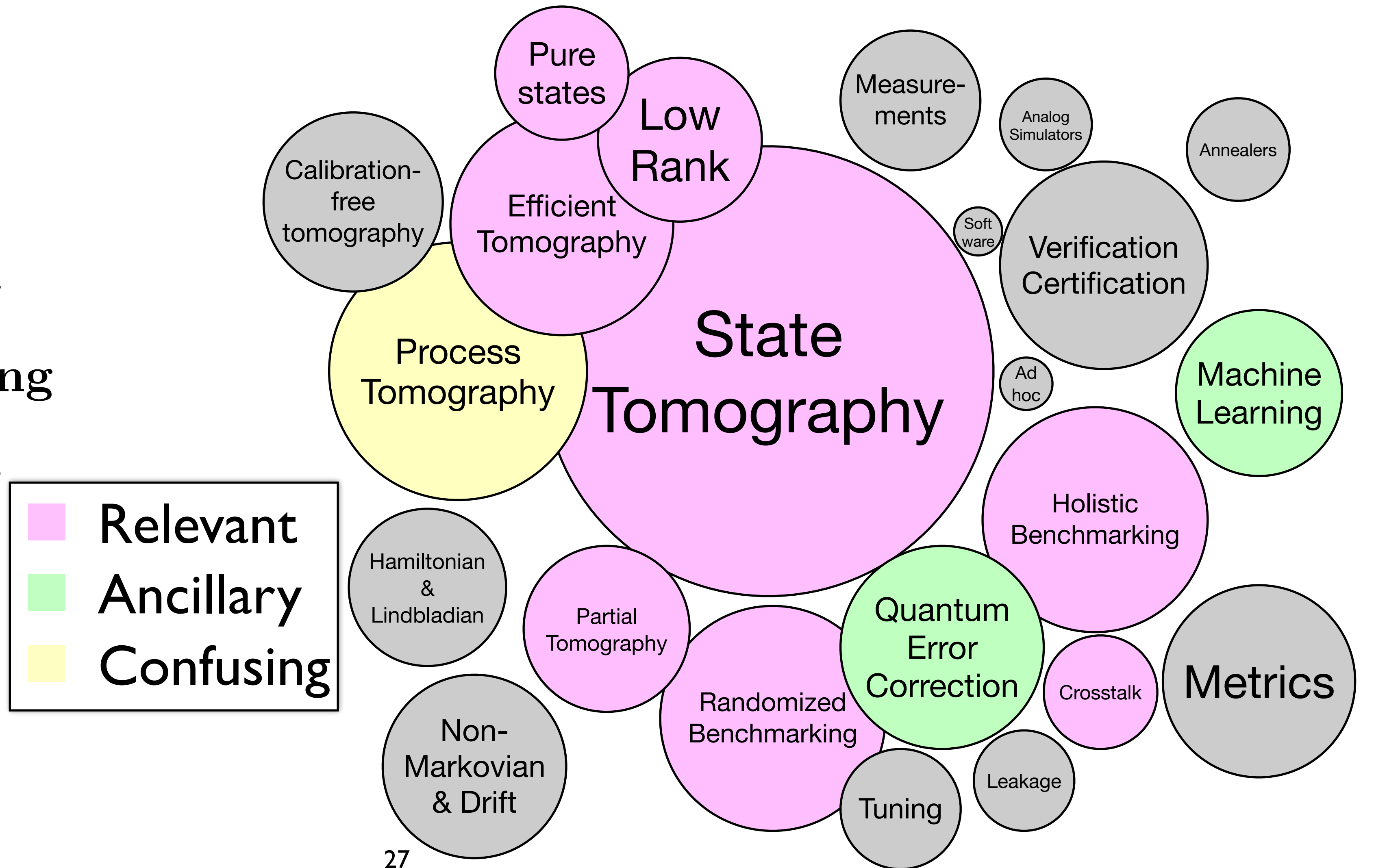
**Randomized benchmarking**

**Direct fidelity estimation**

**Cycle benchmarking**

**X-entropy benchmarking**

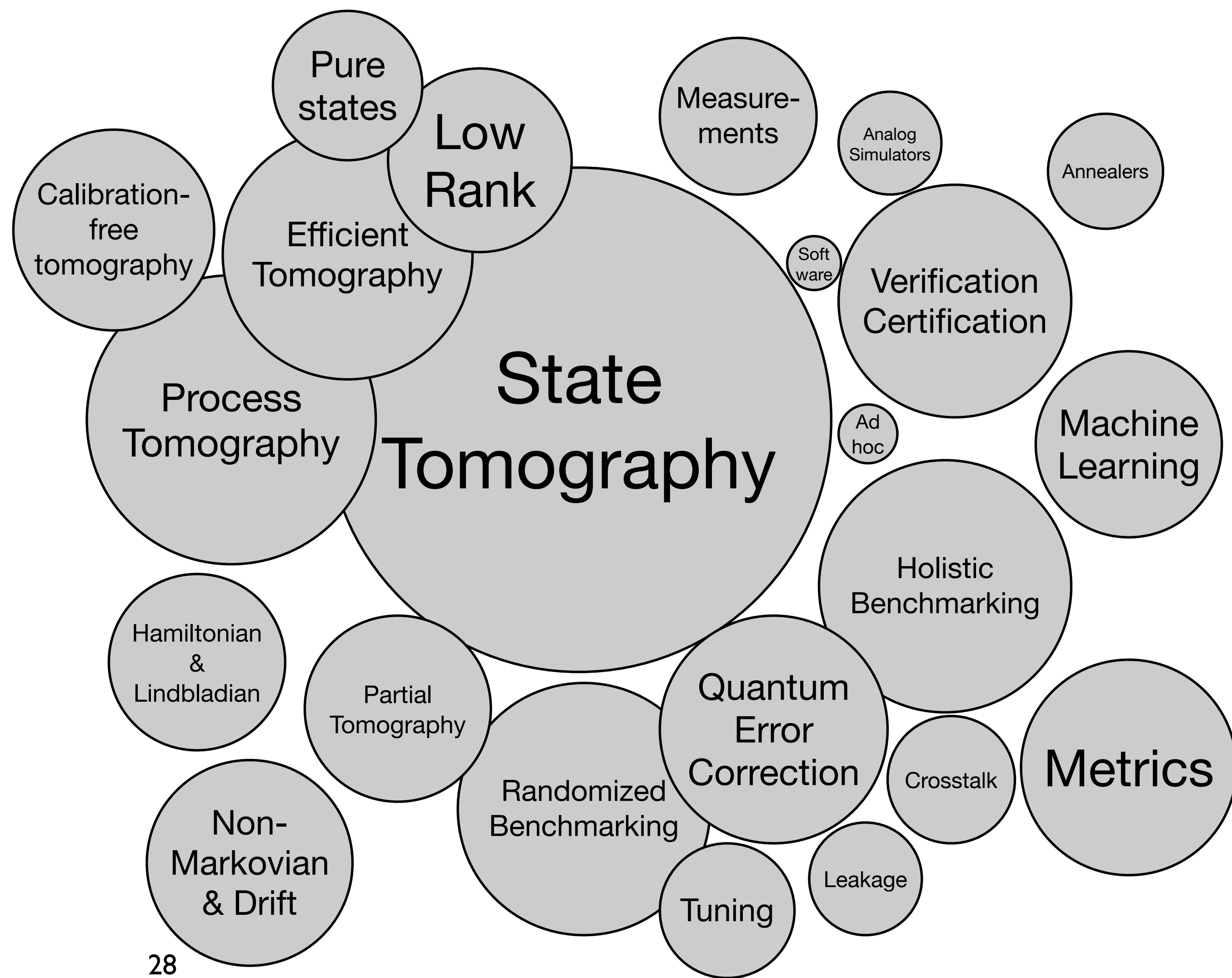
**[Ad hoc QEC stuff]**



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

**Focus:**

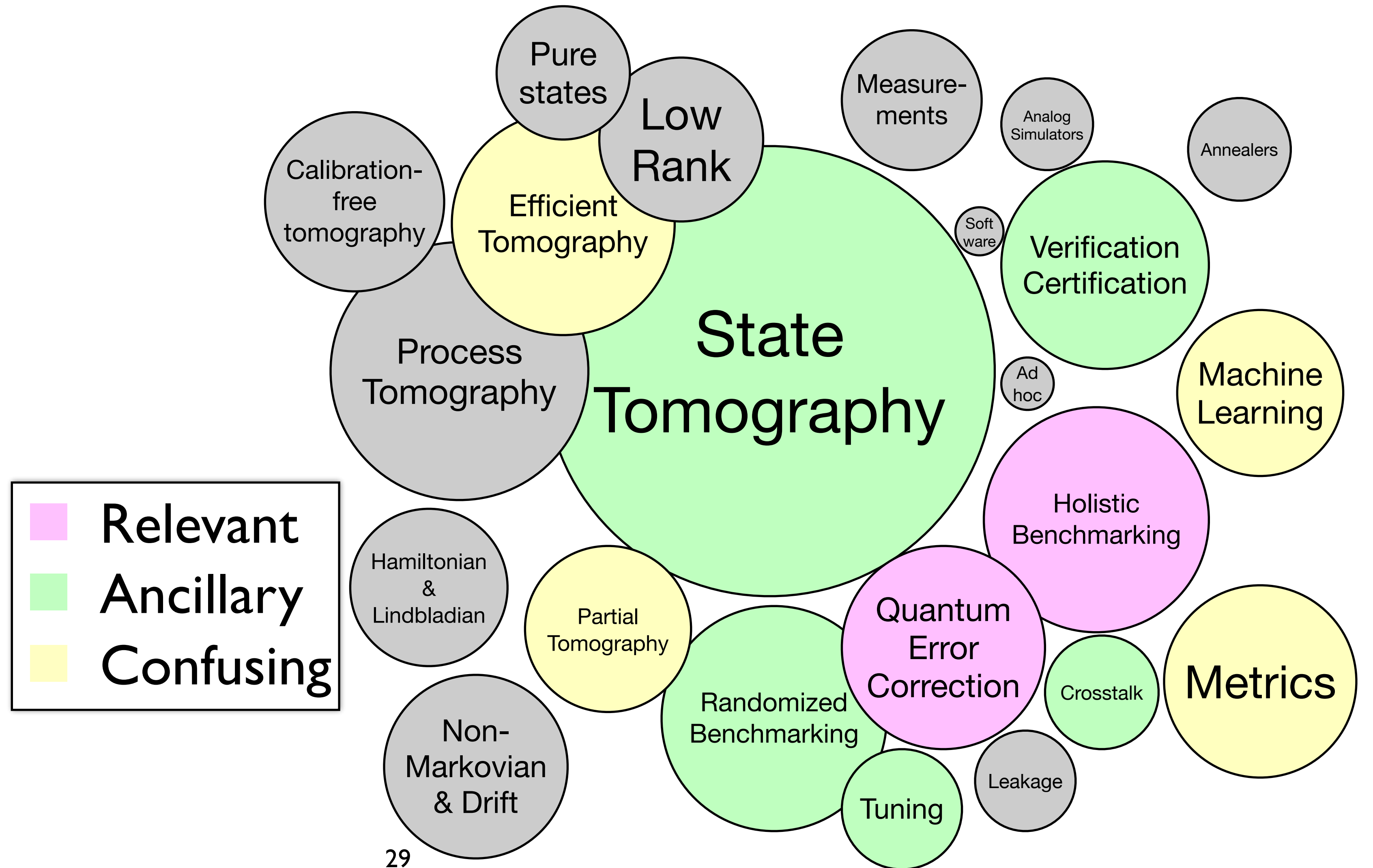
*Benchmarking entire Processors*



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

# Focus:

## *Benchmarking entire Processors*



	Integrated Processors	Composite Components/ Subroutines	Elementary Operations
Testing			
Benchmarking			
Characterization			

**Focus:**

*Benchmarking entire Processors*

State tomography

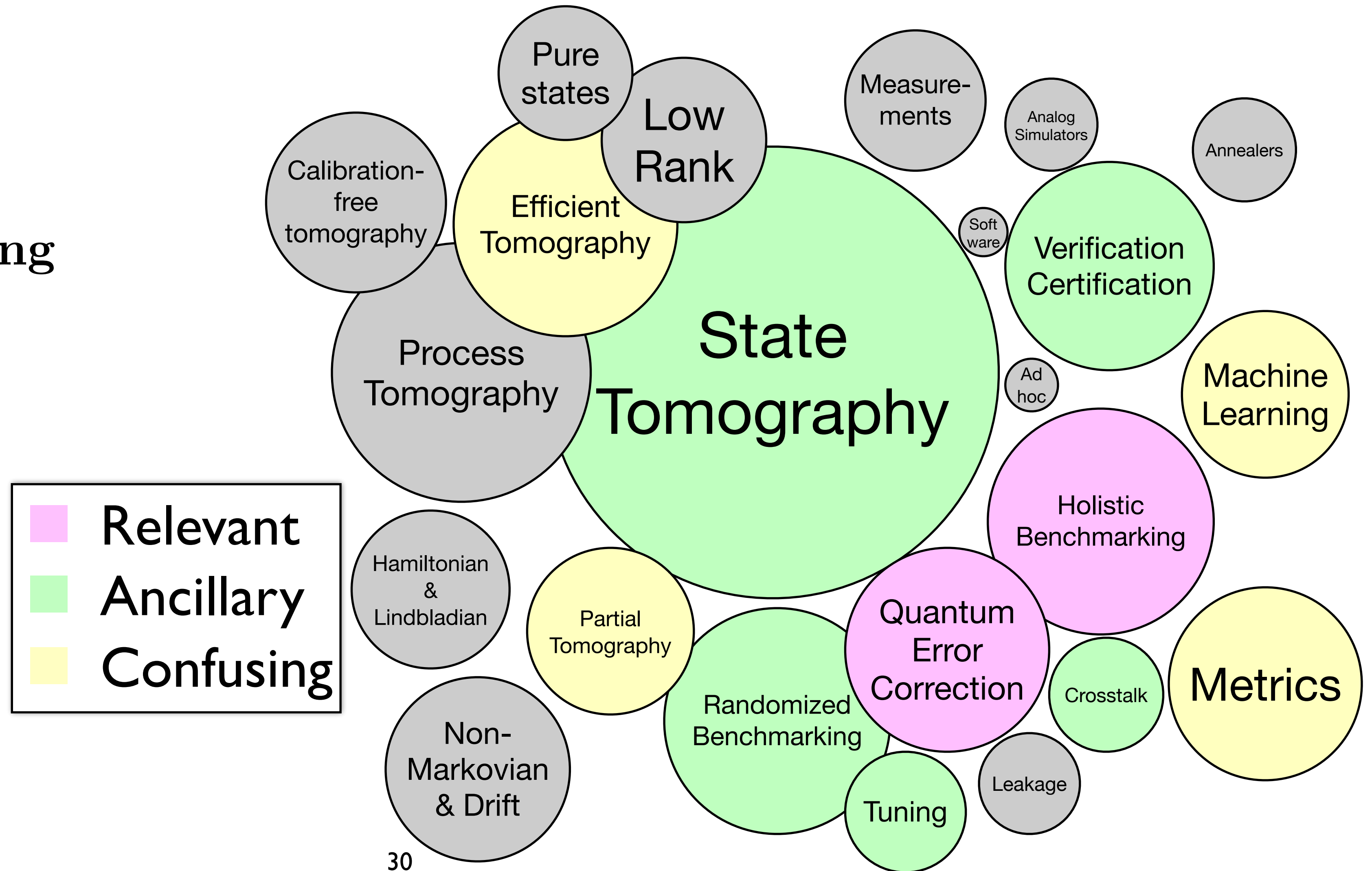
Randomized benchmarking  
- efficient variants

RCS + XEB

Quantum volume

Algorithms

Volumetric benchmarks



# Let's talk about some benchmarks

In the rest of this talk, I'd like to examine the ideas that have been proposed (so far) for “benchmarking” the holistic performance of quantum processors.

I'm not going to try and explain them all in detail!

Mostly, I want to point out why they're all *different*.

And why we need to explore an even more diverse array of benchmarks.

# Let's talk about some benchmarks

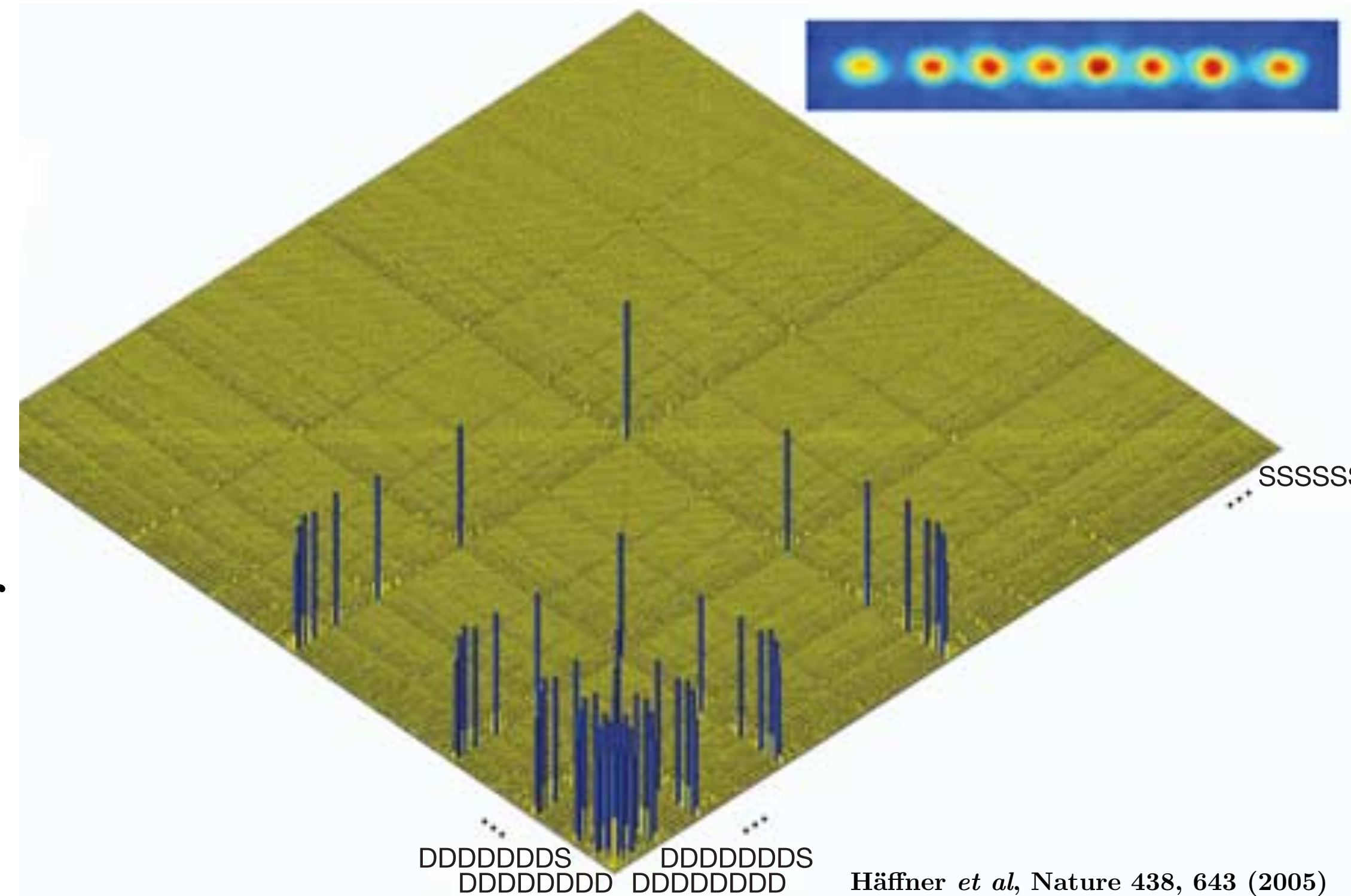
- State tomography
- Randomized benchmarking
- Random circuit sampling (RCS) + cross-entropy benchmarking (XEB)
- Quantum volume
- Algorithms
- Volumetric benchmarks



# State Tomography as a Benchmark

I'm surprised how often state tomography is *still* used in Nature/Science papers.

Idea: demonstrate processor's capability by programming it to produce a particular "interesting" quantum state, and then use state tomography to show that it did so.



This is a weak demo even in theory, since it only demonstrates one circuit. But, in principle, "seeing the state" ought to provide great debugging info.

In practice, neither the state nor the tomography are perfect, so authors end up reporting fidelity " $F = 0.73(4)$ "  $\Rightarrow$  the entire exercise is almost pointless.

# Let's talk about some benchmarks

- State tomography
- Randomized benchmarking
- Random circuit sampling (RCS) + cross-entropy benchmarking (XEB)
- Quantum volume
- Algorithms
- Volumetric benchmarks

# Randomized Benchmarking

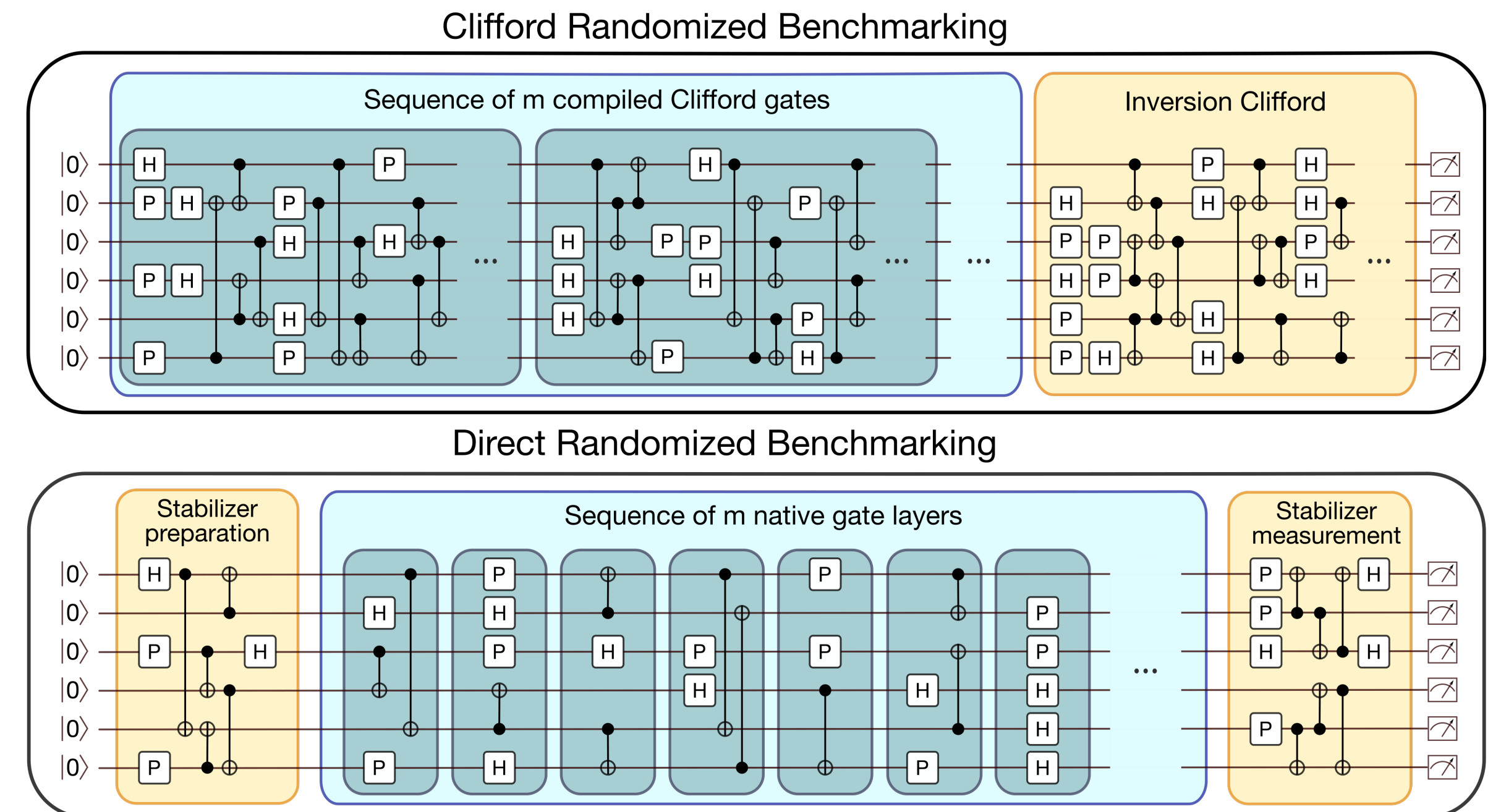
Standard RB runs many random sequences of Clifford operations, followed by a single inversion.

The problem with doing this on  $N$  qubits is that each  $N$ -qubit

Clifford has to be compiled into a *lot* of elementary gates [ $O(N^2)$ ].

For  $N > 3$ , basically every Clifford RB circuit has  $p \approx 0$  of succeeding.

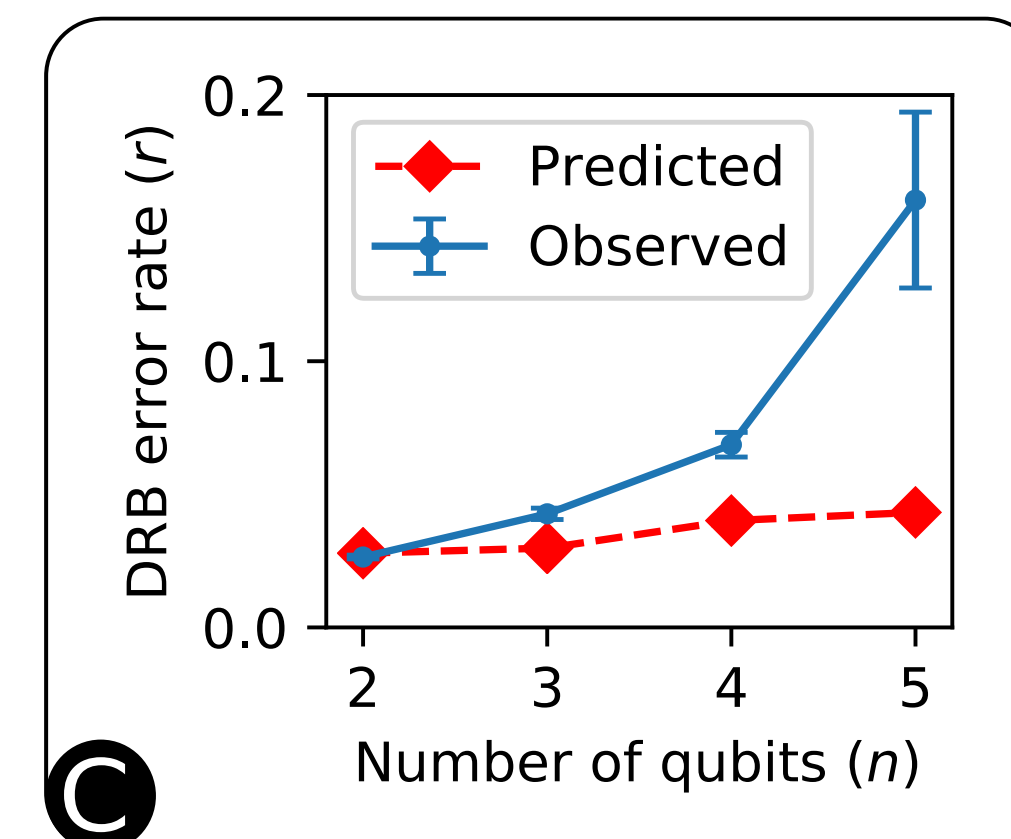
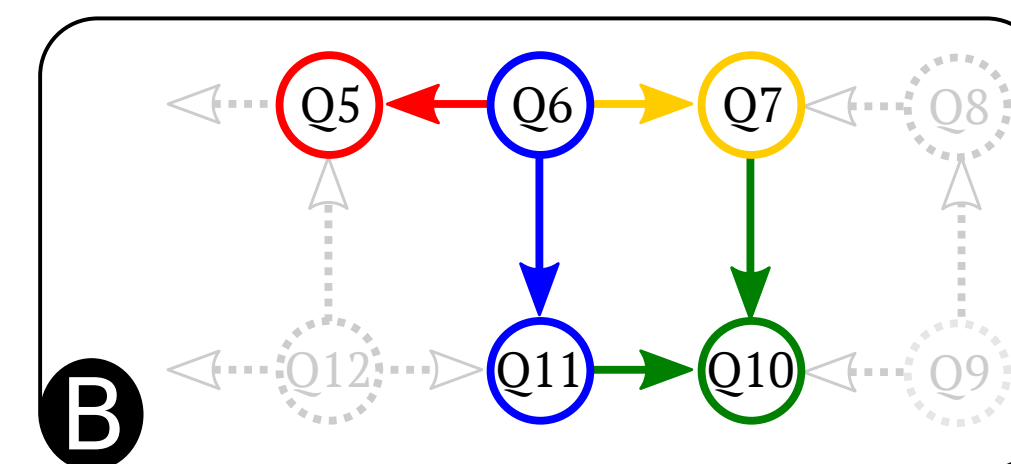
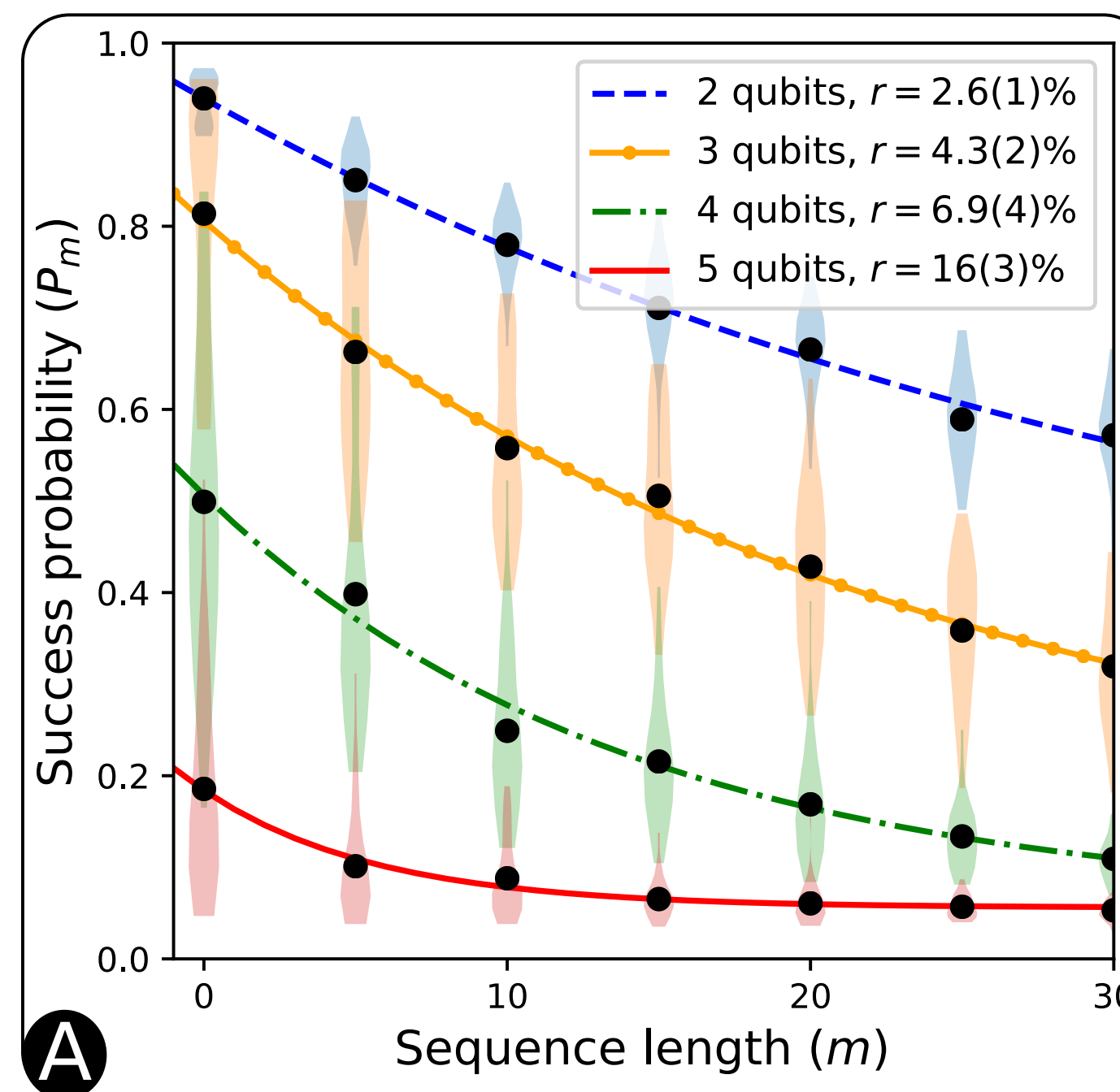
Direct RB is more scalable, but only to about  $N=10$  qubits on current state-of-the-art testbeds.



# Randomized Benchmarking

So what does it measure?

RB (or DRB) measures *the effective error rate per circuit layer in random, unstructured Clifford circuits.*



This is a meaningful, well-motivated quantity.

But it's inextricably linked to the particular ensemble of circuits:

- random, unstructured Clifford-only circuits on all  $N$  qubits.
- depends on how each Clifford is compiled (or relative density of gates)
- only describes *average* behavior — specific circuits can be different.

# Let's talk about some benchmarks

- State tomography
- Randomized benchmarking
- Random circuit sampling (RCS) + cross-entropy benchmarking (XEB)
- Quantum volume
- Algorithms
- Volumetric benchmarks

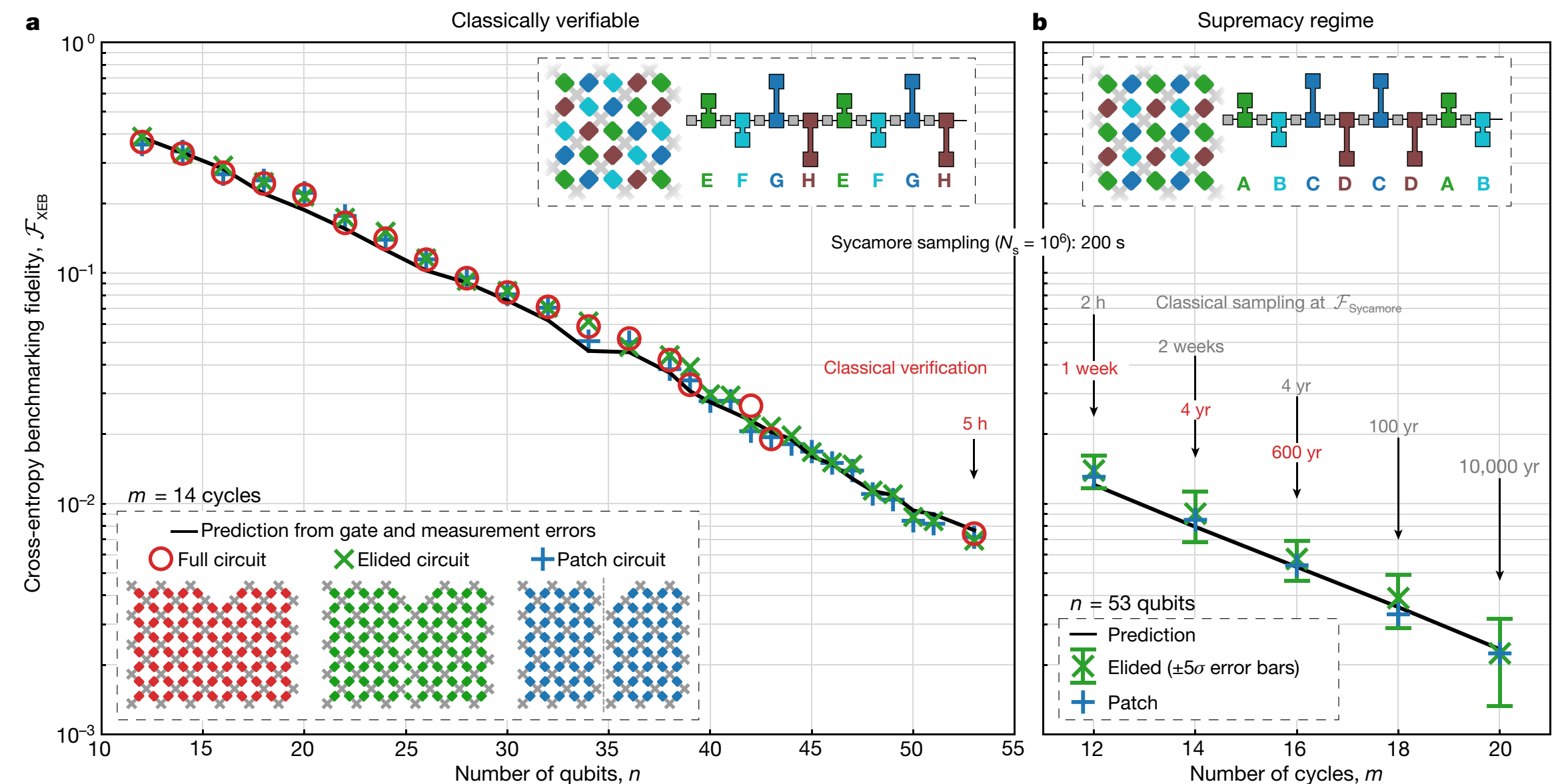
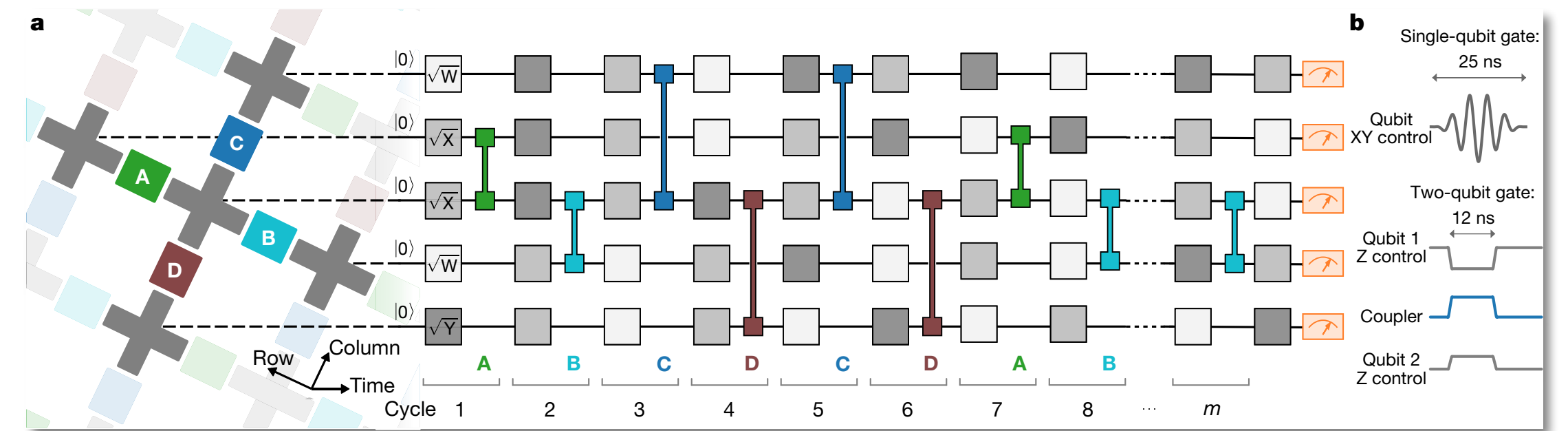
# XEB / RCS

This is what Google used last year to demonstrate quantum supremacy.

Circuits are similar to DRB, but use *non-Clifford* gates, and alternate layers of 1-qubit and 2-qubit gates.

These circuits *scramble* local errors, and (ideally) produce states that look very much like Haar-random ones (unlike RB, there's no inversion step!)

Famously, these are hard to predict.



# Let's talk about some benchmarks

- State tomography
- Randomized benchmarking
- Random circuit sampling (RCS) + cross-entropy benchmarking (XEB)
- Quantum volume
- Algorithms
- Volumetric benchmarks

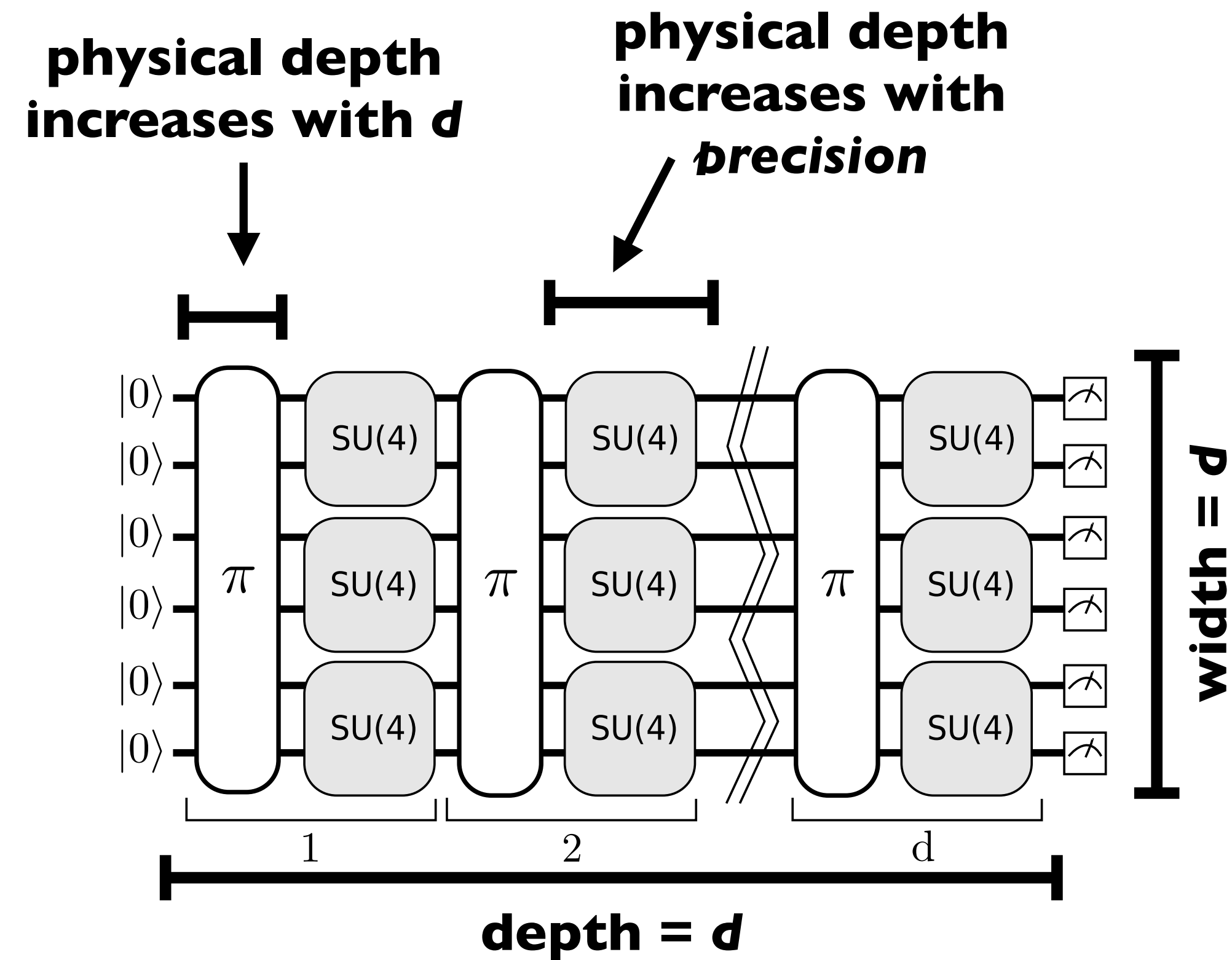
# Quantum Volume

IBM proposed quantum volume as the first *intentional* holistic benchmark.

One number: describes dimension of the Hilbert space that can be accessed uniformly.  
 [ $V$  vs  $\log(V)$  is confusing...]

Specifies yet another class of minimally-structured “random” circuits:

- width and depth vary together as  $d$ ,
- specified by *logical* operations, not *physical* gates
- physical depth can be quite a bit larger than logical!





# Quantum Volume vs RCS / XEB



**Are Google's and IBM's benchmarks measuring the same thing?**

- Both produce random states, and use non-Clifford gates.
- Success metrics (heavy-output probability and “linear cross-entropy”) are nominally different, but actually closely related and roughly equivalent).

QV circuits are highly compilation-dependent (RCS are not).

QV circuits are much deeper (for given width) than you might expect.

RCS allows independently varying width/depth.

RCS is *very* good at scrambling errors; QV depends on compilation.

# Let's talk about some benchmarks

- State tomography
- Randomized benchmarking
- Random circuit sampling (RCS) + cross-entropy benchmarking (XEB)
- Quantum volume
- Algorithms
- Volumetric benchmarks

# Algorithms as a Benchmark

Classical benchmarks are usually a basket of real-world tasks.

There's a long history of running [tiny] algorithms on quantum processors.

In principle, this is a great idea:

- We expect algorithm circuits to behave different from random ones.
- Algorithms would properly test the *whole* system including compiler.
- Running algorithms would allow benchmarking quantum vs classical!

In practice, it's premature:

- Testbed processors can't run any meaningful algorithms yet.
- Algorithms and compilers are changing too fast for stable benchmarks.
- Most promising algorithms are *hybrid* and don't just test the quantum processor.

# Algorithms as a Benchmark

Benchmarks based on algorithms / applications are clearly the future!

But we need to develop a systematic framework for building them, starting by asking “What aspects of performance do we want to measure?”

Quantum Volume is an nice early step *toward* this — it attempts to synthesize several distinct and important properties (error rate, error type, connectivity, programmability, compiler efficiency) in a single number.

But we need to understand (and benchmark) these properties *separately* too, before we can intelligently mix them to simulate application performance.

**We need more study of what impacts algorithm performance.**

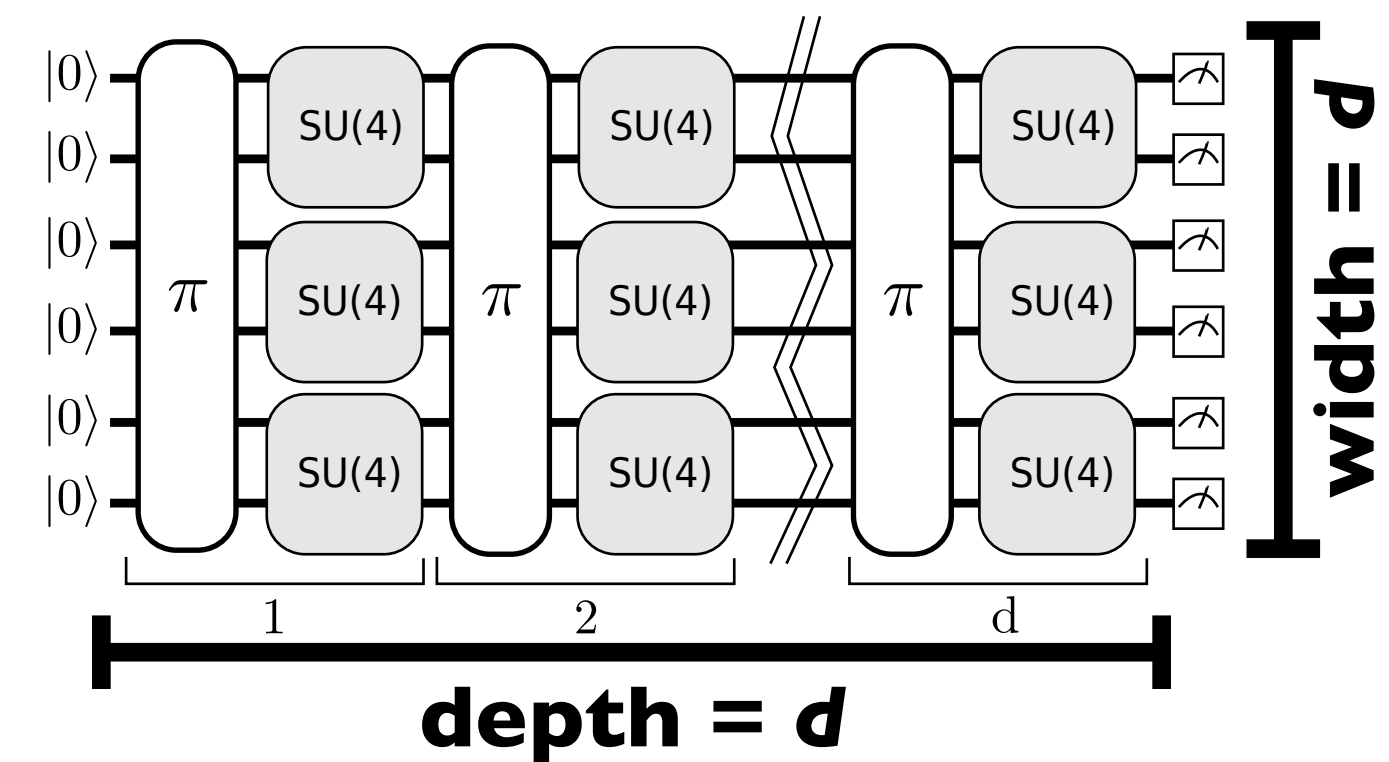
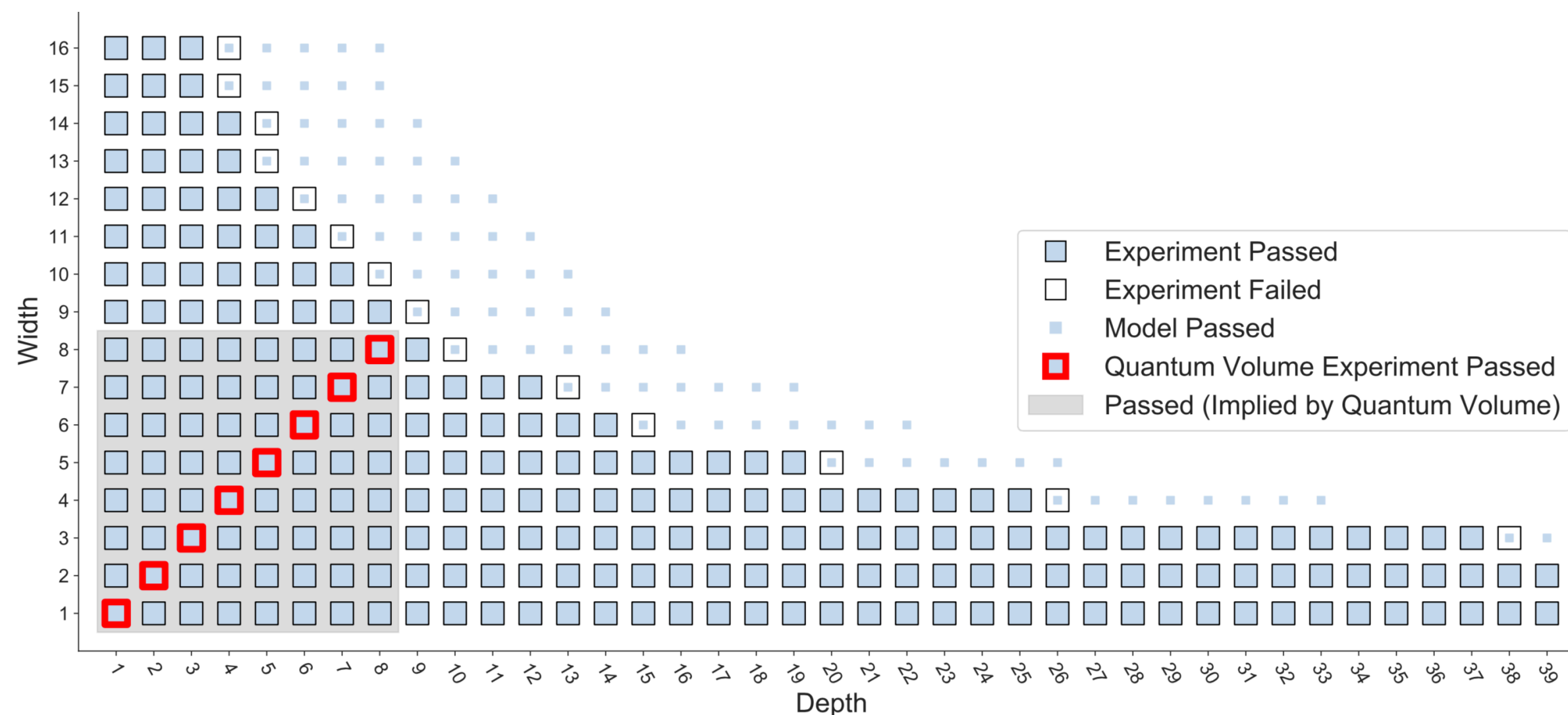
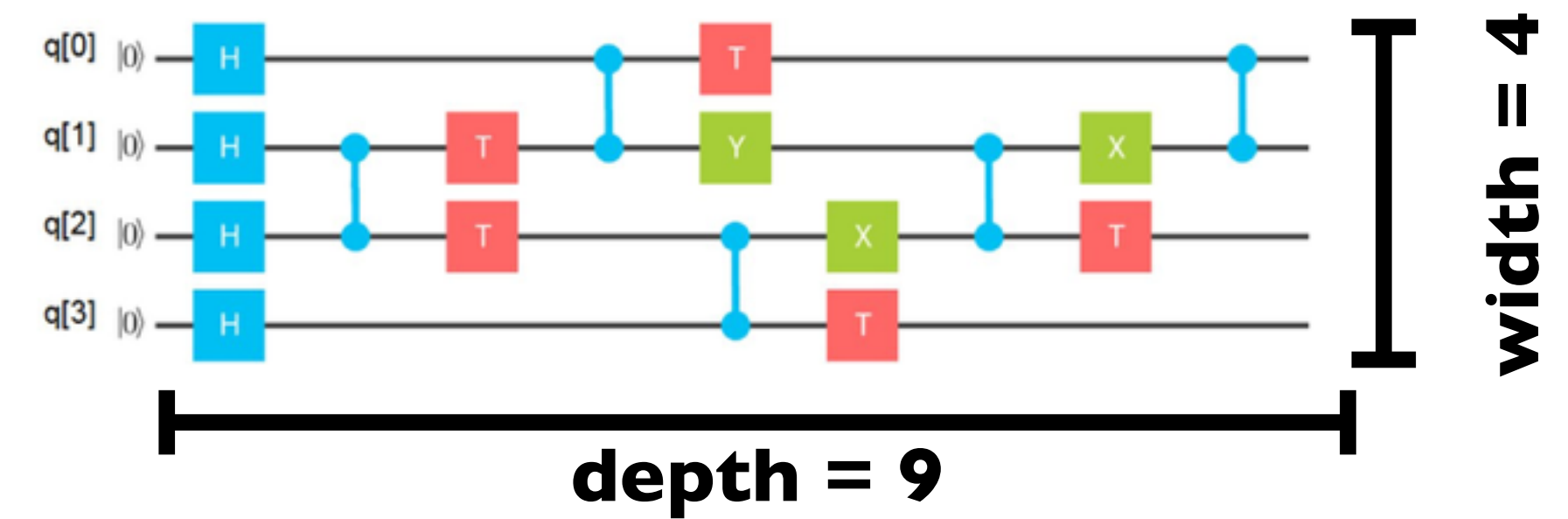
# Let's talk about some benchmarks

- State tomography
- Randomized benchmarking
- Random circuit sampling (RCS) + cross-entropy benchmarking (XEB)
- Quantum volume
- Algorithms
- Volumetric benchmarks

# Volumetric Benchmarks

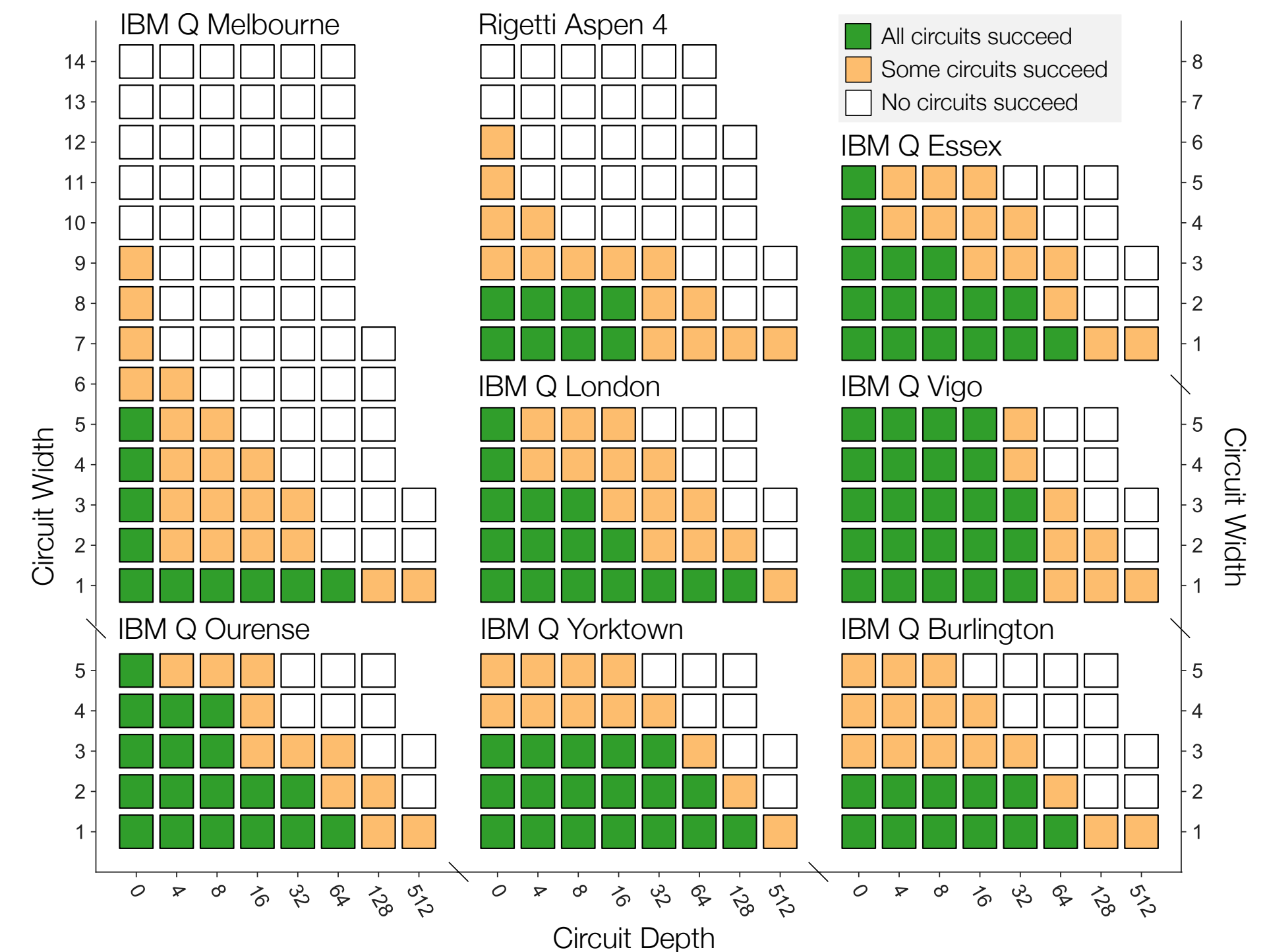
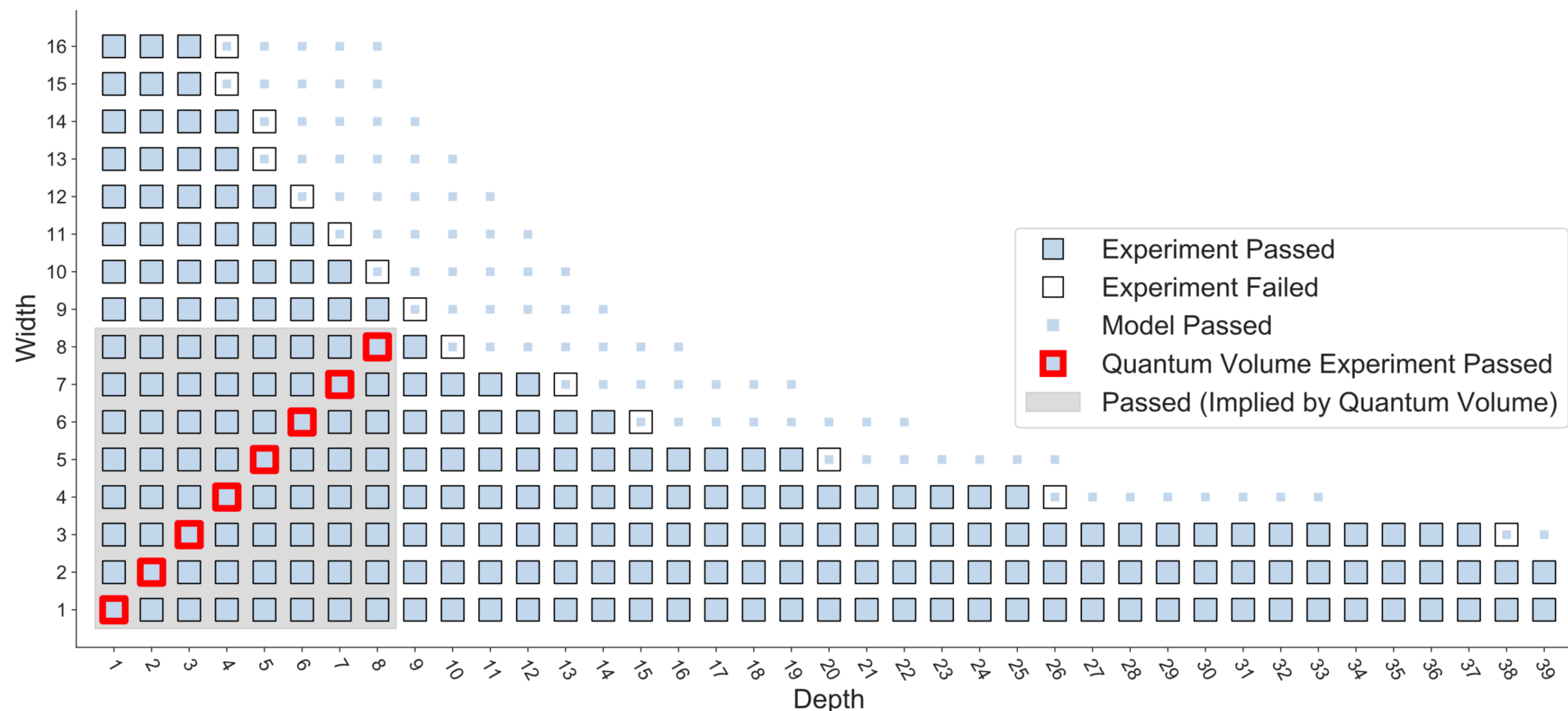
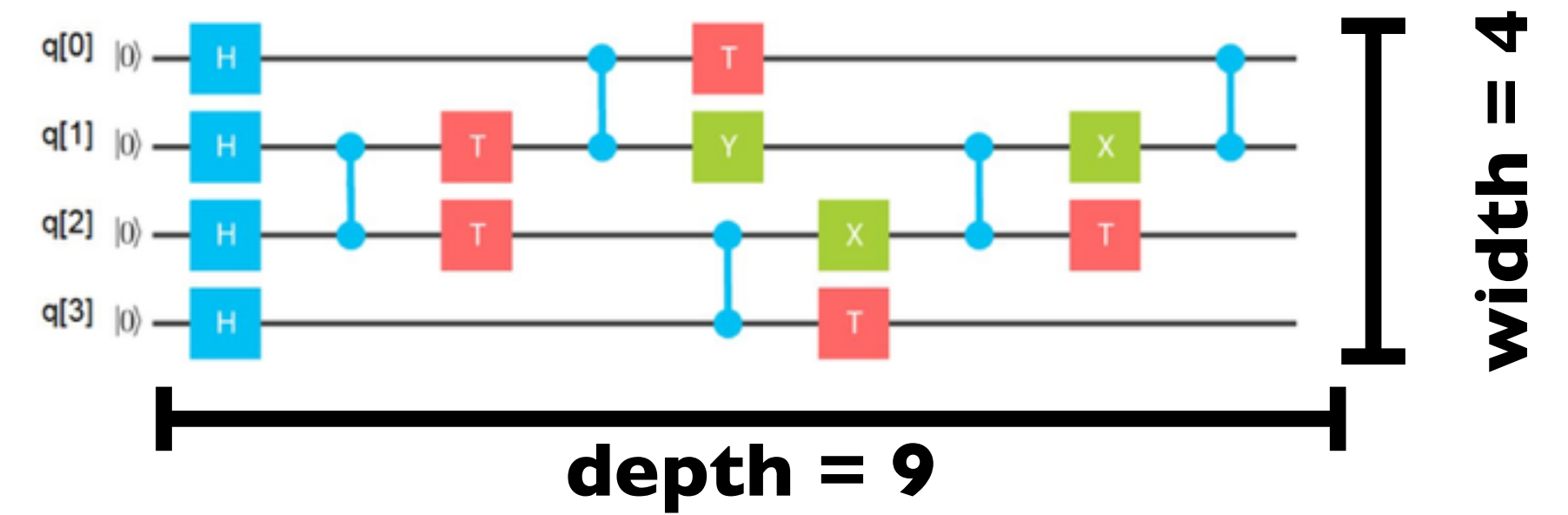
- In many circumstances, what we want to know about a quantum processor is:
 

“What is its *capability* — what circuits can it run with high success probability?”
- This is largely determined by three properties of the circuit:
  - *width* (number of qubits)
  - *depth* (number of clock cycles)
  - *structure* (what kind of circuit is it?)
- Quantum volume measures ability to run random square (width = depth) circuits.



# Volumetric Benchmarks

- Quantum volume seeks a single-number answer
- But volume usually doesn't capture the full range of a processor's capability — or limitations.
- We introduced *volumetric benchmarks* to measure ability to run *all* circuit shapes, of *any* structure.



# Going forward: Measuring QC performance

Benchmarking and characterization remain separate (and complementary)

We are going to need them both: characterization of low-level properties to *predict* performance on circuits, and benchmarks to *measure* it.

*(Eventually, these had better converge...)*

We are rapidly developing new ideas about how to measure and benchmark the overall performance of quantum processors!

There have been major discrete advances just in the past few years.

We need *more* creative benchmarks — and we need them with clear statements about “**What, exactly, does this benchmark measure?**”