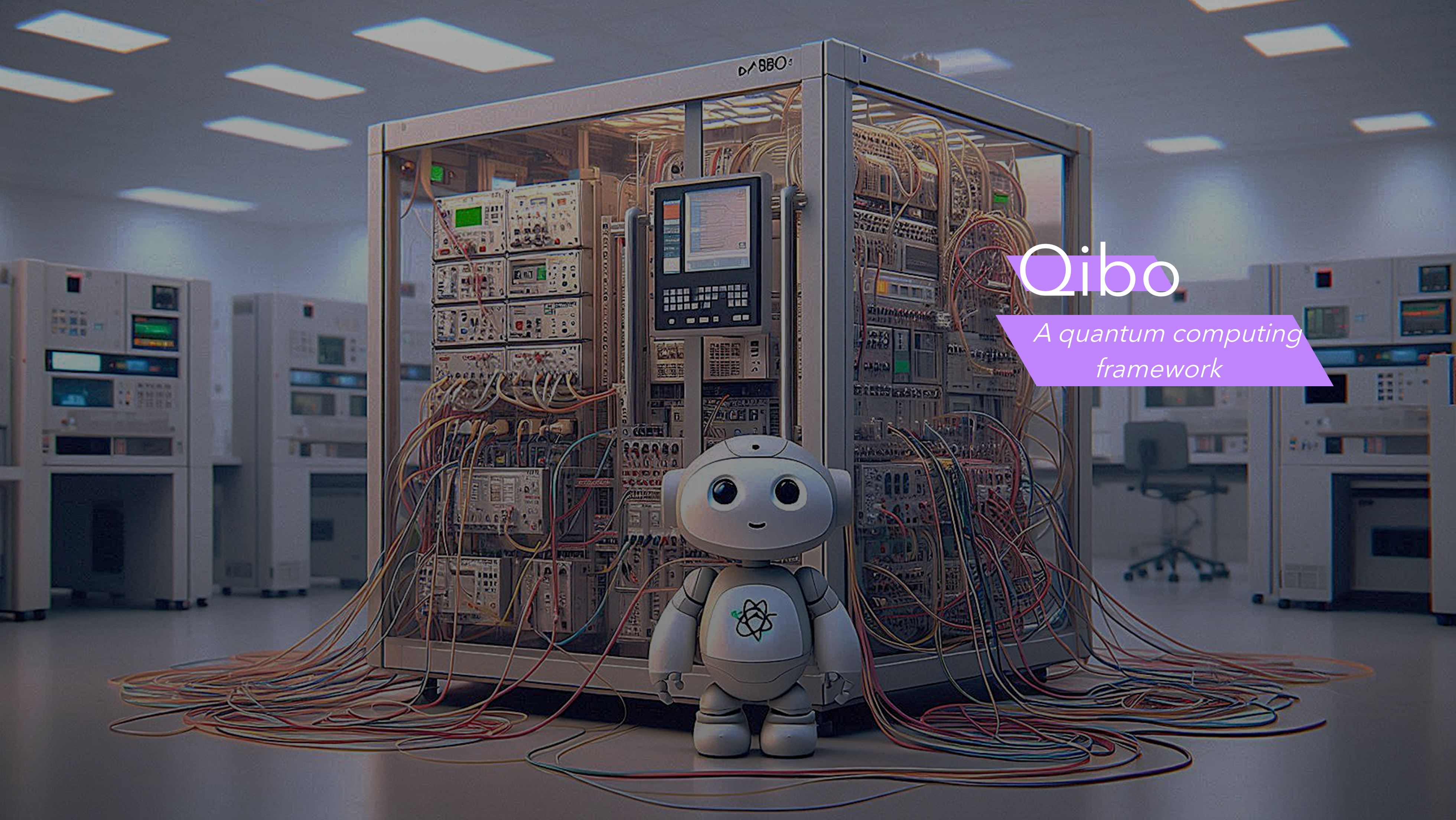


Qibo

A quantum computing framework

A bit of context…

R&D and adoption of new technologies in HEP

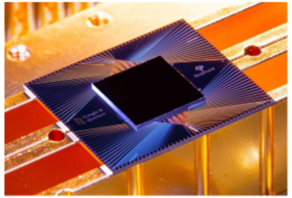
HEP is moving towards new technologies, in particular hardware accelerators

CPU

GPU

FPGA/ASIC

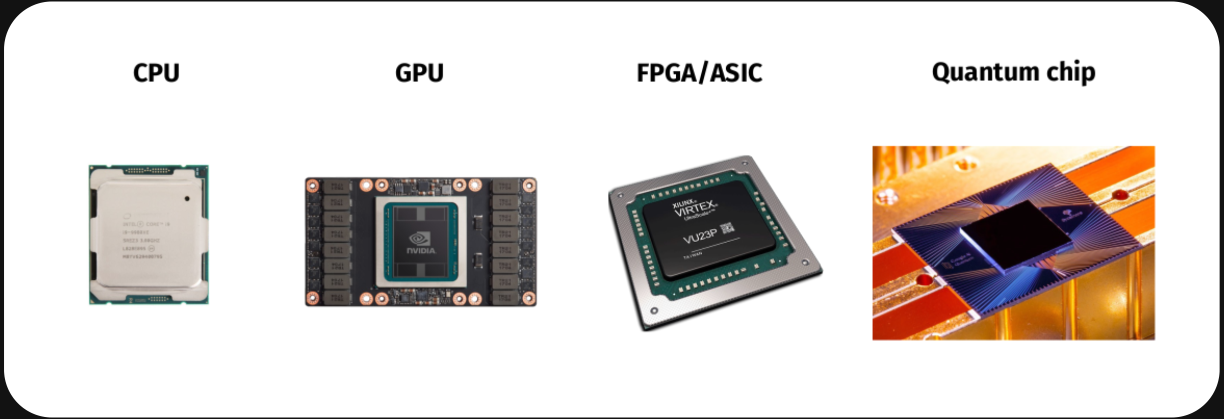
Quantum chip



Moving from general purpose devices ⇒ application specific

R&D and adoption of new technologies in HEP

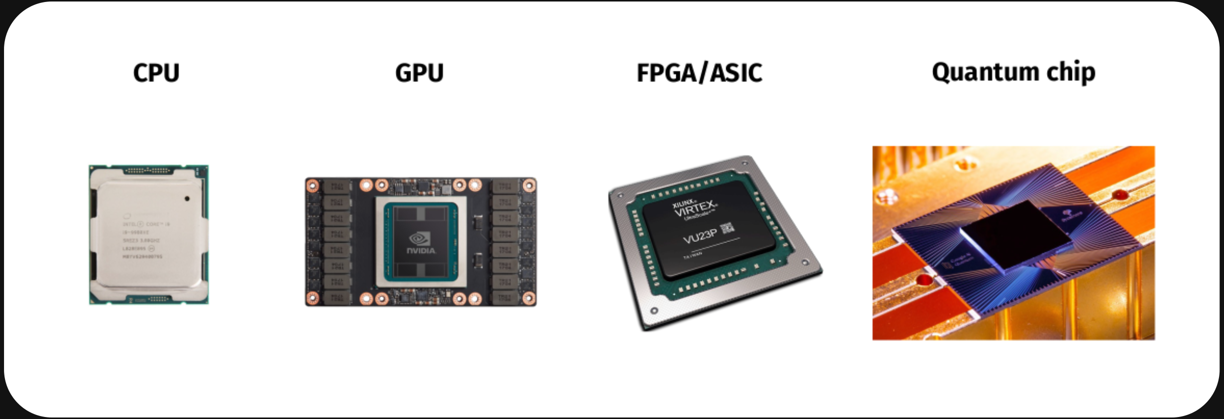
HEP is moving towards new technologies, in particular hardware accelerators



Moving from general purpose devices ⇒ application specific

R&D and adoption of new technologies in HEP

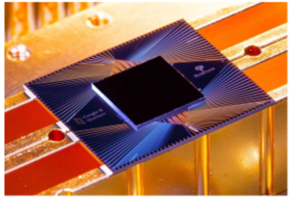
HEP is moving towards new technologies, in particular hardware accelerators



Moving from general purpose devices ⇒ application specific

R&D and adoption of new technologies in HEP

HEP is moving towards new technologies, in particular hardware accelerators

CPU	GPU	FPGA/ASIC	Quantum chip
			

Moving from general purpose devices ⇒ application specific

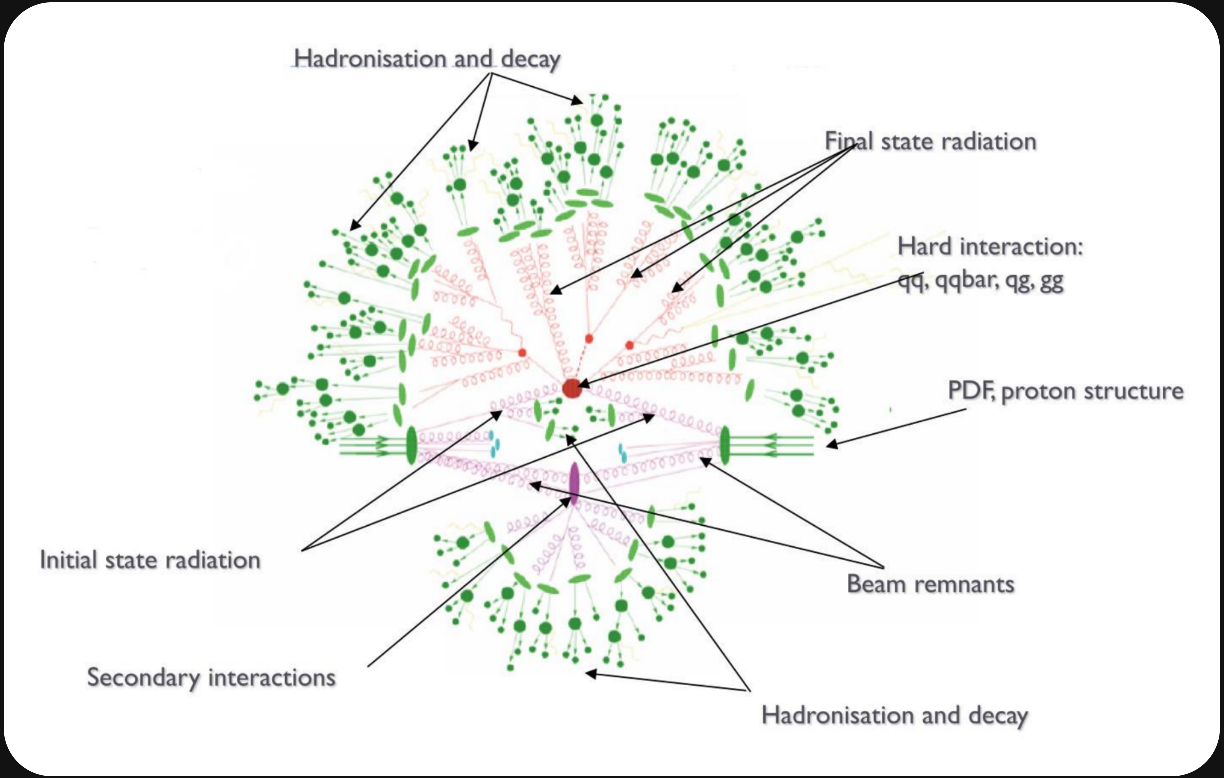
Examples of initiatives and institutions involved:

CERN | QUANTUM TECHNOLOGY INITIATIVE

INFN

BIQUTE

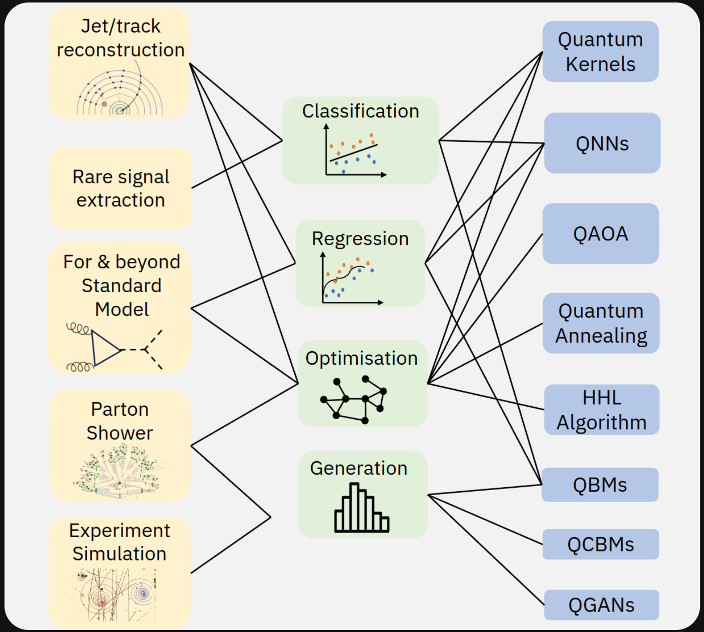
E.g.

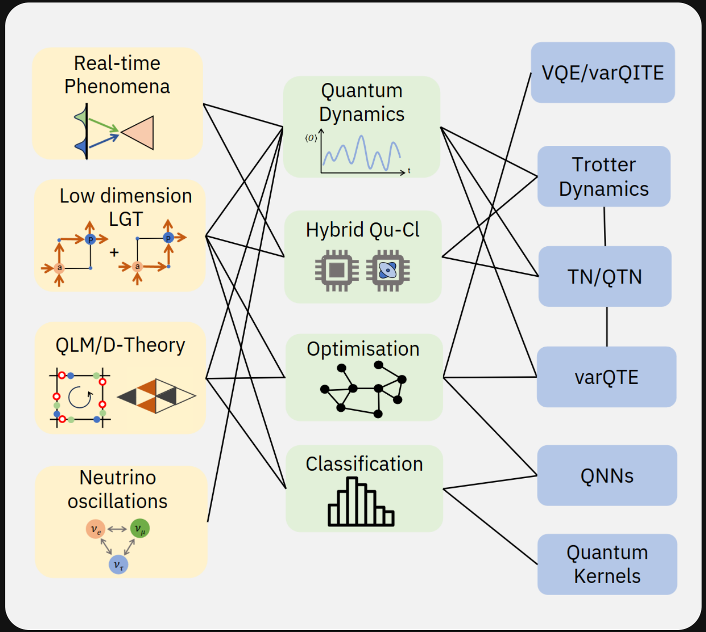


Monte Carlo simulation and data analysis are intensive and requires lots of computing power.

Quantum computing for HEP experiments

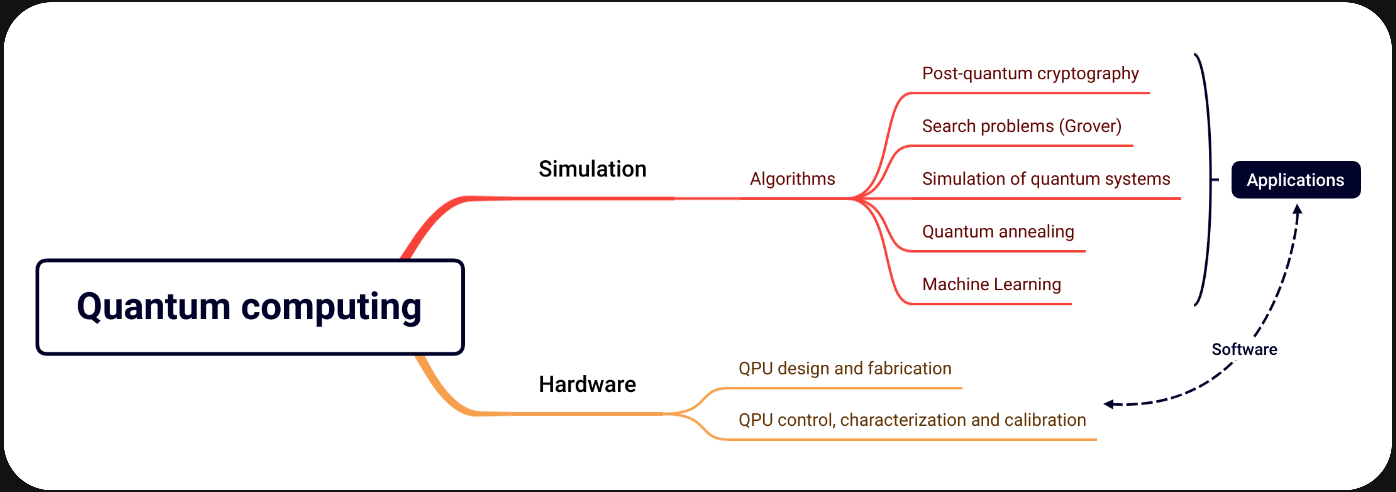
QC4HEP WG `[arXiv: 2307.03236]`





Many experimental and theoretical HEP applications are deemed to benefit from quantum computation.

Recap



Simulation

- required to develop algorithms
- complete introspection
- require noise modeling

Hardware

- limited (in many senses)
- requires calibration
- final validation

Discrete gates primer

Goal: Construct a generic $U(2^n)$ operation based on building blocks

The Hilbert space on which the unitaries act is a strutured as a $\bigotimes$ tensor product of n qubits

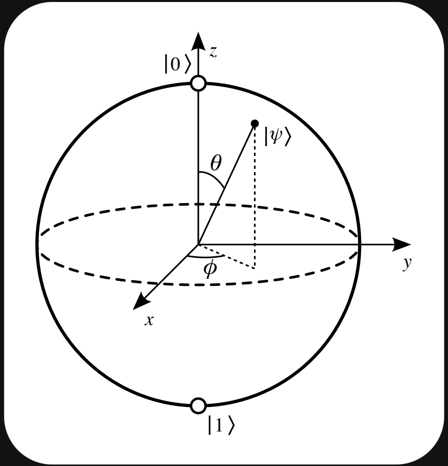
$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

the generic qubit state is:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \qquad \text{with } |\alpha|^2 + |\beta|^2 = 1$$

and it can be visualized as a point on the Bloch sphere

$$\alpha = \cos\theta/2 \qquad \beta = e^{-i\phi}\sin\theta/2$$



Example gates: Pauli

X gate

Z gate

The X gate acts like the classical NOT gate, it is represented by the σ_x matrix,

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

therefore

$$|0\rangle \longrightarrow |1\rangle$$
$$|1\rangle \longrightarrow |0\rangle$$

The Z gate flips the sign of $|1\rangle$, it is represented by the σ_z matrix,

$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

therefore

$$|0\rangle \longrightarrow |0\rangle$$
$$|1\rangle \longrightarrow -|1\rangle$$

Single-qubit gates

These are operations on the Bloch sphere

Two-qubit gates

The building-block interactions

Multi-qubit gates

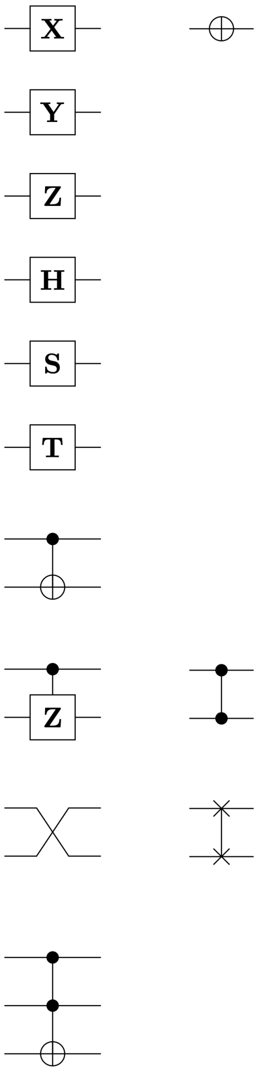
Higher-level instructions for algorithms

Define a universal gate set

- **universality** means it can generate all unitarities
- possibly **redundant**, since it may be efficient to execute
- **multiple** implementations, related to diverse hardware

Gates could be variously parametrized, so there exists universal sets made beyond

Gate	Matrix
Pauli-X (X)	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y (Y)	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z (Z)	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard (H)	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Phase (S, P)	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
$\pi/8$ (T)	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$
Controlled Not (CNOT, CX)	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
Controlled Z (CZ)	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$
SWAP	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Toffoli (CCNOT, CCX, TOFF)	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$



Single-qubit gates

These are operations on the Bloch sphere

Two-qubit gates

The building-block interactions

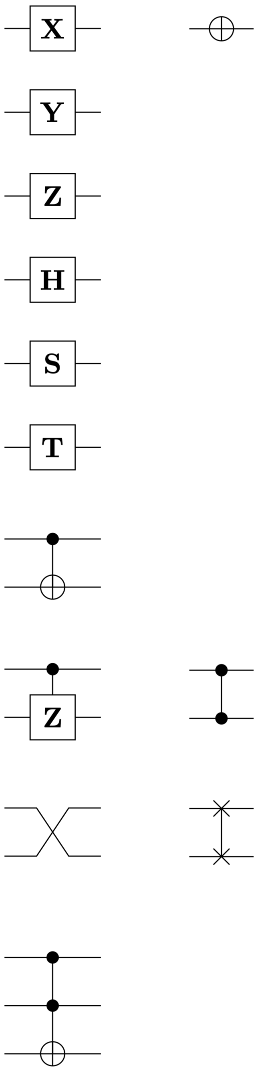
Multi-qubit gates

Higher-level instructions for algorithms

Define a universal gate set

- **universality** means it can generate all unitarities
- possibly **redundant**, since it may be efficient to execute
- **multiple** implementations, related to diverse hardware

Gates could be variously parametrized, so there exists universal sets made beyond

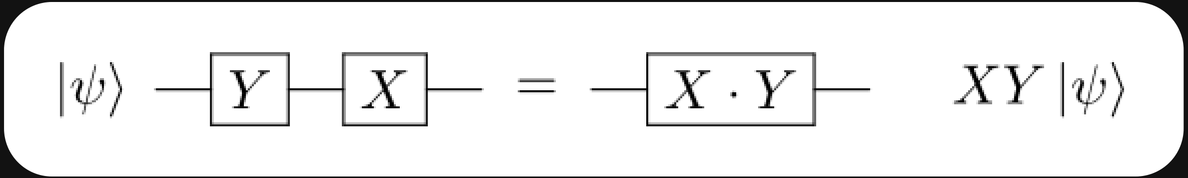


Pauli-X (X)	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$
Pauli-Y (Y)	$\begin{bmatrix}0 & -i\\i & 0\end{bmatrix}$
Pauli-Z (Z)	$\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}$
Hadamard (H)	$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}$
Phase (S, P)	$\begin{bmatrix}1 & 0\\0 & i\end{bmatrix}$
$\pi/8$ (T)	$\begin{bmatrix}1 & 0\\0 & e^{i\pi/4}\end{bmatrix}$
Controlled Not (CNOT, CX)	$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$
Controlled Z (CZ)	$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&-1\end{bmatrix}$
SWAP	$\begin{bmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{bmatrix}$
Toffoli (CCNOT, CCX, TOFF)	$\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&1&0\end{bmatrix}$

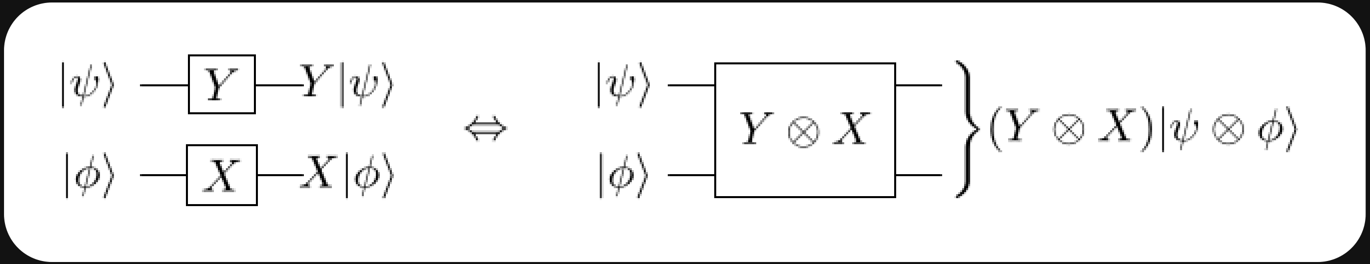
Circuit

Unitary - but measurements.

Circuit are a way to compose gates to build unitaries, ***sequentially***



or in ***parallel***



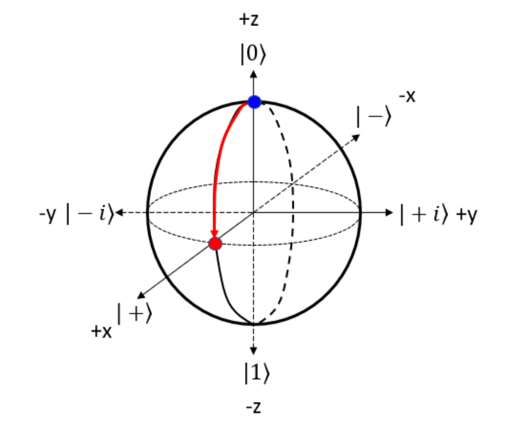
Parametrized gate

Rotations gates (Bloch sphere)

$$R_y(\theta) \equiv e^{-i\theta\frac{\sigma_y}{2}} = \begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix}$$

Note that $R_y(\pi) \equiv Y$.

Every unitary transformation as decomposed in rotations (*Euler's angles*)



Other parameters are possible: GPI and $GPI2$ parametrize the position of the axis, multi-qubit gates can paramterize complex interactions, …

Having parameters, it opens the door to optimization 🚀 → i.e. quantum machine learning (QML)

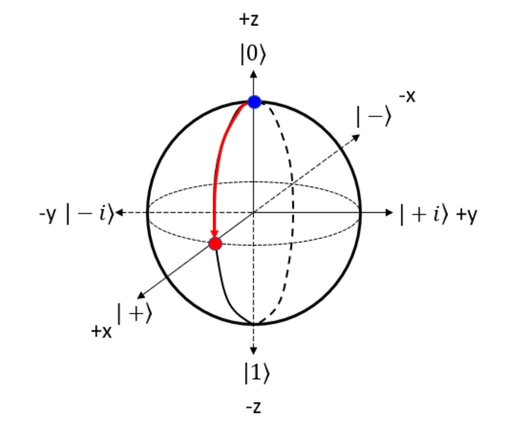
Parametrized gate

Rotations gates (Bloch sphere)

$$R_y(\theta) \equiv e^{-i\theta\frac{\sigma_y}{2}} = \begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix}$$

Note that $R_y(\pi) \equiv Y$.

Every unitary transformation as decomposed in rotations (*Euler's angles*)



Other parameters are possible: GPI and $GPI2$ parametrize the position of the axis, multi-qubit gates can paramterize complex interactions, …

Having parameters, it opens the door to optimization 🚀 → i.e. quantum machine learning (QML)

Two-qubit gate

The atoms of interaction

Controlled gates (conditionals)

The controlled-NOT ($CNOT$) gate is a conditional gate defined as

$$CNOT \equiv \begin{pmatrix} 1 & 0 \\ 0 & \sigma_x \end{pmatrix}$$

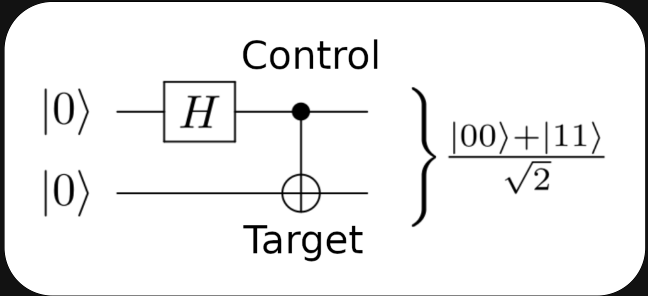
We define a control qubit which, if at $|1\rangle$, applies X to a target qubit.

control
target

$|00\rangle \rightarrow |00\rangle$ $\quad$ $|01\rangle \rightarrow |01\rangle$

$|10\rangle \rightarrow |11\rangle$ $\quad$ $|11\rangle \rightarrow |10\rangle$

Multi-qubit gates allow entangling states



Measurement

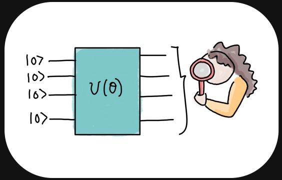
The non-unitary gate *that you have*

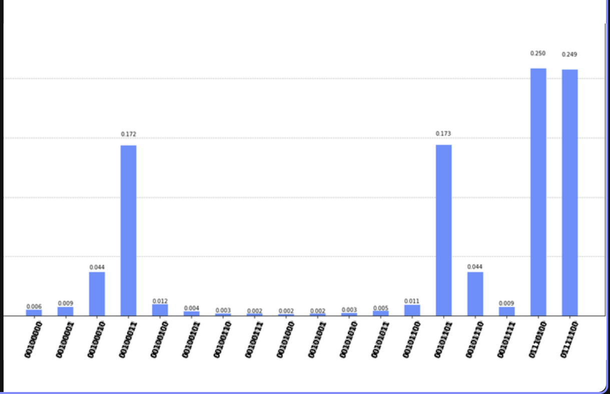
Measurements are special gates, in two ways:

1. it is the only operation that allows to extract information
2. it is the only non-unitary gate

Shots

(Module of) amplitudes of the final states are derived by repeating the experiment many times identically, performing many *shots*.





Noise and channels

Non-unitary operations model

Instead of acting over a state vector, the state will be tracked by a density matrix

$$|\psi\rangle \quad \longrightarrow \quad \rho \quad (\sim |\psi\rangle \langle\psi| \,)$$

This makes possible to track phenomena like decoherence, which has not a unitary action on the state.

Another option is to exploit measurement non-unitarity, and represent the noise through *repeated execution*.

- Kraus

$$\Phi(\rho) = \sum_i B_i \rho B_i^*$$

- Stinespring

$$U_0 = \sum_\alpha K_\alpha \otimes |\alpha\rangle \langle v_0|$$

- Choi

$$\Lambda = |U\rangle\rangle\langle\langle U|$$

- Liouville, Quantum networks, …

Noise and channels

Non-unitary operations model

Instead of acting over a state vector, the state will be tracked by a density matrix

$$|\psi\rangle \quad \longrightarrow \quad \rho \quad (\sim |\psi\rangle\langle\psi|)$$

This makes possible to track phenomena like decoherence, which has not a unitary action on the state.

Another option is to exploit measurement non-unitarity, and represent the noise through *repeated execution*.

- Kraus

$$\Phi(\rho) = \sum_i B_i \rho B_i^*$$

- Stinespring

$$U_0 = \sum_\alpha K_\alpha \otimes |\alpha\rangle\langle v_0|$$

- Choi

$$\Lambda = |U\rangle\rangle\langle\langle U|$$

- Liouville, Quantum networks, …

Noise and channels

Non-unitary operations model

Instead of acting over a state vector, the state will be tracked by a density matrix

$$|\psi\rangle \quad \longrightarrow \quad \rho \quad (\sim |\psi\rangle \langle\psi|)$$

This makes possible to track phenomena like decoherence, which has not a unitary action on the state.

Another option is to exploit measurement non-unitarity, and represent the noise through *repeated execution*.

- Kraus

$$\Phi(\rho) = \sum_i B_i \rho B_i^*$$

- Stinespring

$$U_0 = \sum_\alpha K_\alpha \otimes |\alpha\rangle \langle v_0|$$

- Choi

$$\Lambda = |U\rangle\rangle\langle\langle U|$$

- Liouville, Quantum networks, …

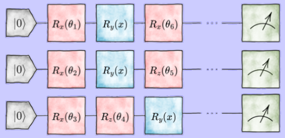
Applications

Quantum machine learning

Machine Learning
$\mathcal{M}$: model;
$\mathcal{O}$: optimizer;
$\mathcal{J}$: loss function.
(x, y): data

Quantum Computation
$\mathcal{Q}$: qubits;
$\mathcal{S}$: superposition;
$\mathcal{E}$: entanglement.

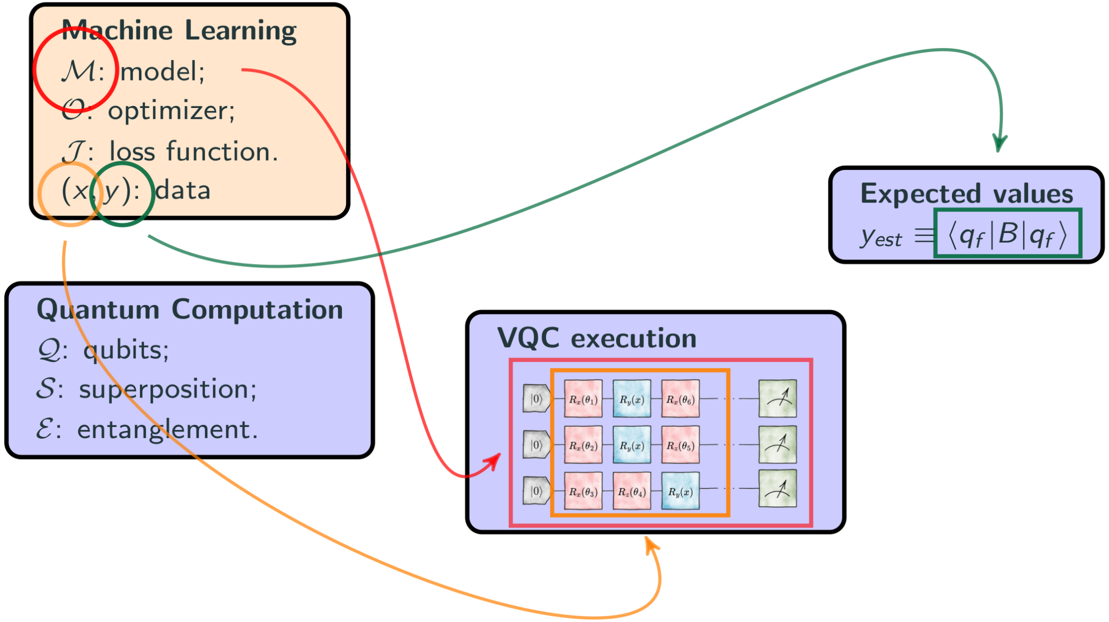
VQC execution



Expected values
$y_{est} \equiv \langle q_f | B | q_f \rangle$

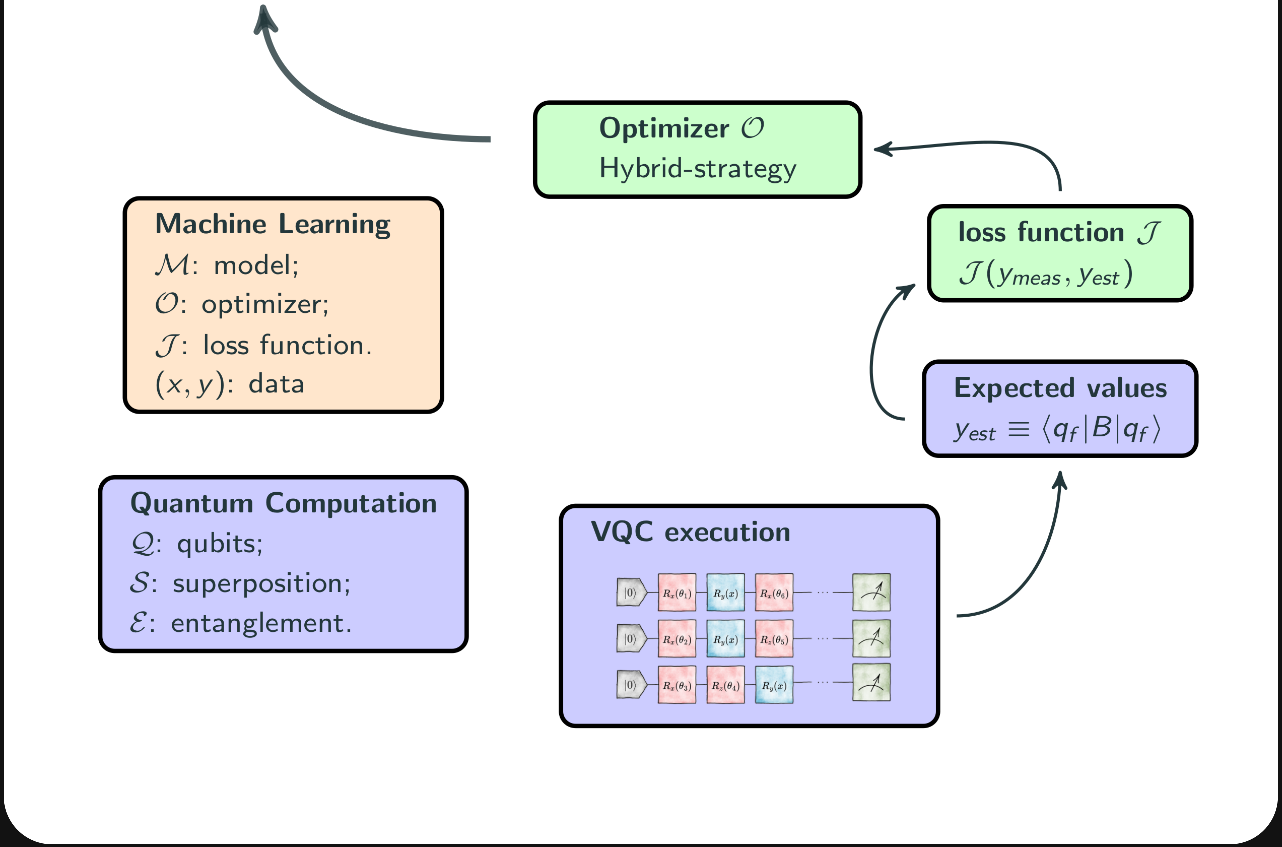
credits M. Robbiati

Quantum machine learning



credits M. Robbiati

Quantum machine learning



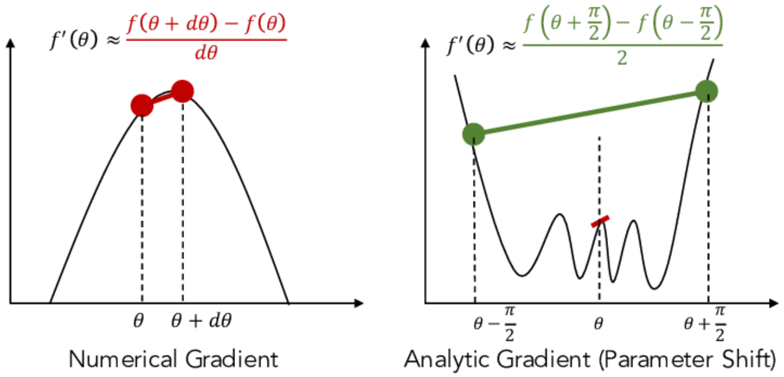
credits M. Robbiati

QML - remarks

A classical function being clasically optimized.

$$\bar{y}_{est}(\bar{\theta}) = \langle 0 | \, U(\bar{\theta}) \, | 0 \rangle \quad : \quad \mathbb{R}^n \rightarrow \mathbb{R}^m$$

If a first-order optimization 🚀 method used, gradient calculation may be "quantum-aware" (PSR). →



The advantage is mainly in the inference time, and possibly ansatz expressivity.

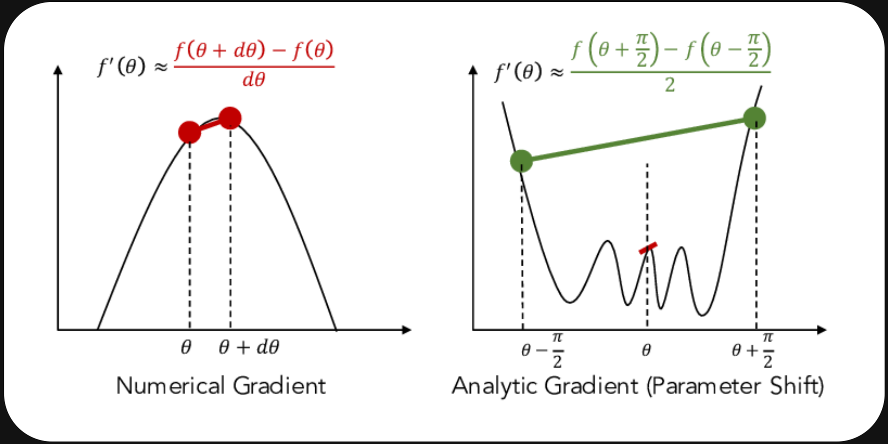
Quantum computation is naturally based on continuous variables. But in practice they are generated through digital control electronics with noisy calibrated pulses

QML - remarks

A classical function being clasically optimized.

$$\bar{y}_{est}(\bar{\theta}) = \langle 0 | U(\bar{\theta}) | 0 \rangle \quad : \quad \mathbb{R}^n \rightarrow \mathbb{R}^m$$

If a first-order optimization 🚀 method used, gradient calculation may be "quantum-aware" (PSR). →



The advantage is mainly in the inference time, and possibly ansatz expressivity.

Quantum computation is naturally based on continuous variables. But in practice they are generated through digital control electronics with noisy calibrated pulses

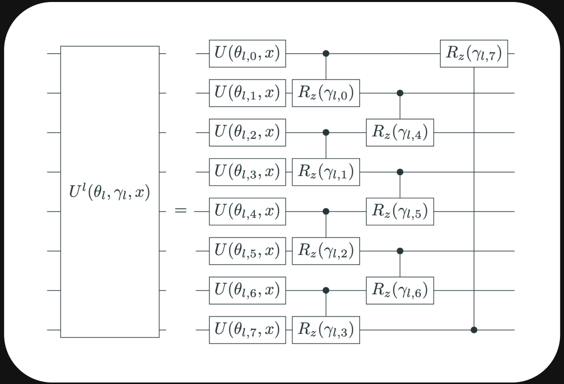
qPDF [arXiv: 2011.13934]

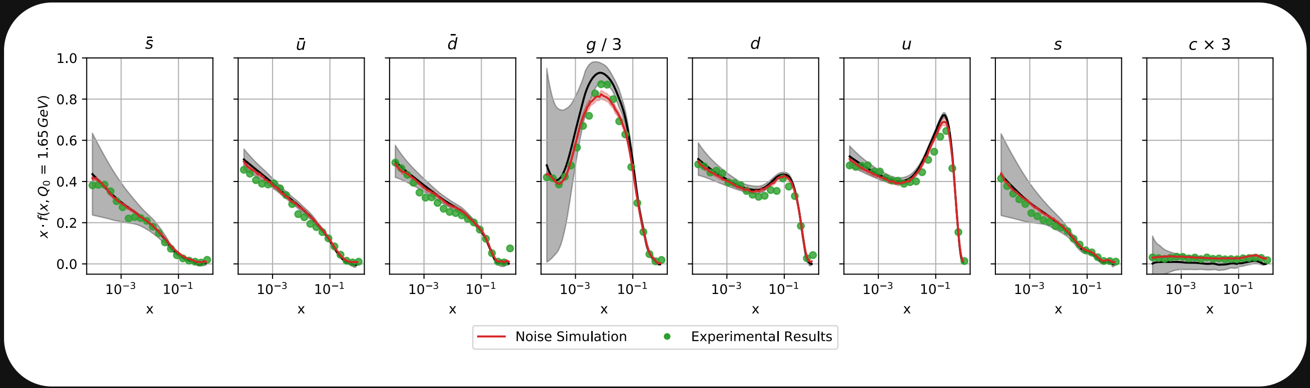
Parametrize ***Parton Distribution Functions (PDF)*** with multi-qubit variational quantum circuits

Algorithm's summary:

1. Define a quantum circuit: $\mathcal{U}(\theta, x)\,|0\rangle^{\otimes n} = |\psi(\theta, x)\rangle$
2. $\mathcal{U}_w(\alpha, x) = R_z(\alpha_3 \log(x) + \alpha_4) R_z(\alpha_1 \log(x) + \alpha_2)$
3. Using $z_i(\theta, x) = \langle \psi(\theta, x) |\, Z_i \,| \psi(\theta, x) \rangle$

$$\mathrm{qPDF}_i(x, Q_0, \theta) = \frac{1 - z_i(\theta, x)}{1 + z_i(\theta, x)}$$





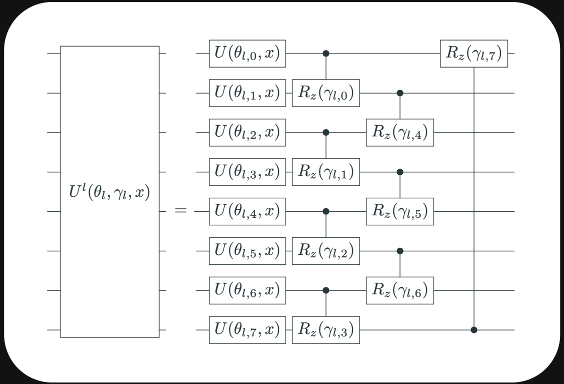
qPDF [arXiv: 2011.13934]

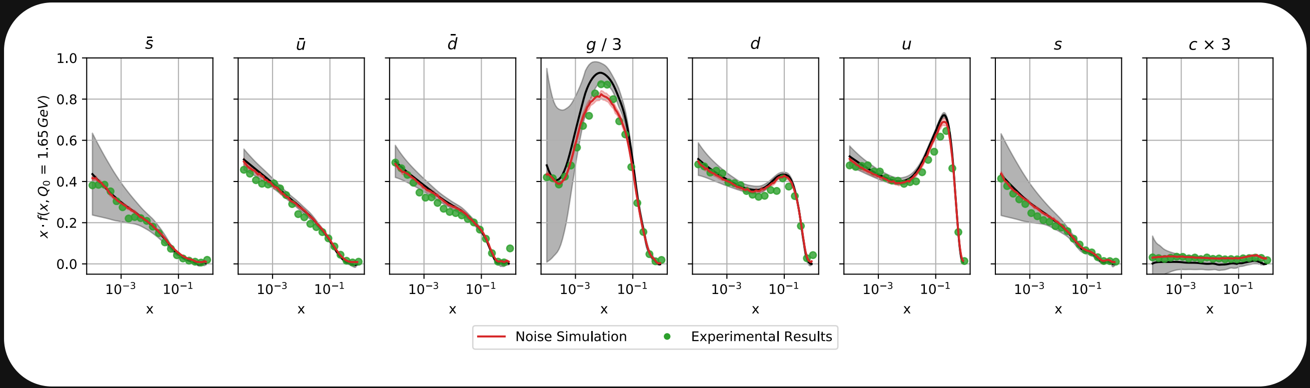
Parametrize ***Parton Distribution Functions (PDF)*** with multi-qubit variational quantum circuits

Algorithm's summary:

1. Define a quantum circuit: $\mathcal{U}(\theta, x)\,|0\rangle^{\otimes n} = |\psi(\theta, x)\rangle$
2. $\mathcal{U}_w(\alpha, x) = R_z(\alpha_3 \log(x) + \alpha_4) R_z(\alpha_1 \log(x) + \alpha_2)$
3. Using $z_i(\theta, x) = \langle\psi(\theta, x)|\, Z_i \,|\psi(\theta, x)\rangle$

$$\mathrm{qPDF}_i(x, Q_0, \theta) = \frac{1 - z_i(\theta, x)}{1 + z_i(\theta, x)}$$





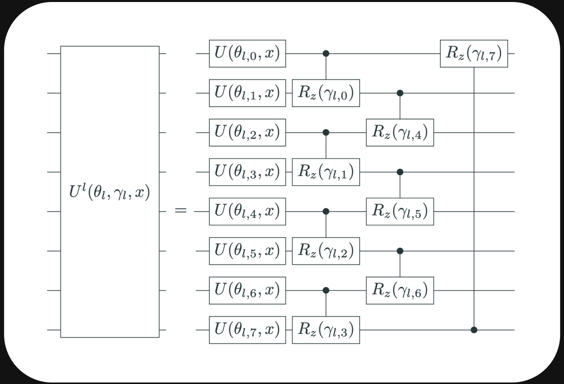
qPDF [arXiv: 2011.13934]

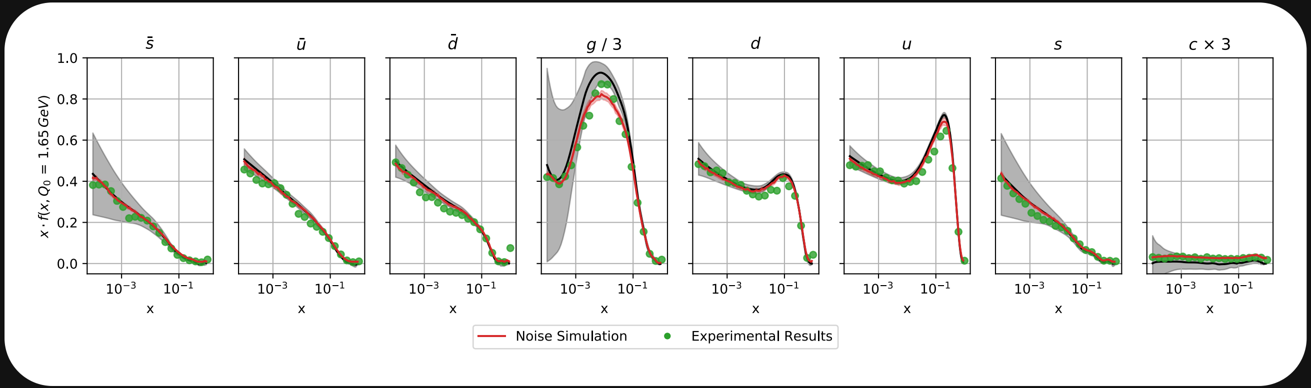
Parametrize ***Parton Distribution Functions (PDF)*** with multi-qubit variational quantum circuits

Algorithm's summary:

1. Define a quantum circuit: $\mathcal{U}(\theta, x) |0\rangle^{\otimes n} = |\psi(\theta, x)\rangle$
2. $\mathcal{U}_w(\alpha, x) = R_z(\alpha_3 \log(x) + \alpha_4) R_z(\alpha_1 \log(x) + \alpha_2)$
3. Using $z_i(\theta, x) = \langle \psi(\theta, x) | Z_i | \psi(\theta, x) \rangle$

$$\text{qPDF}_i(x, Q_0, \theta) = \frac{1 - z_i(\theta, x)}{1 + z_i(\theta, x)}$$





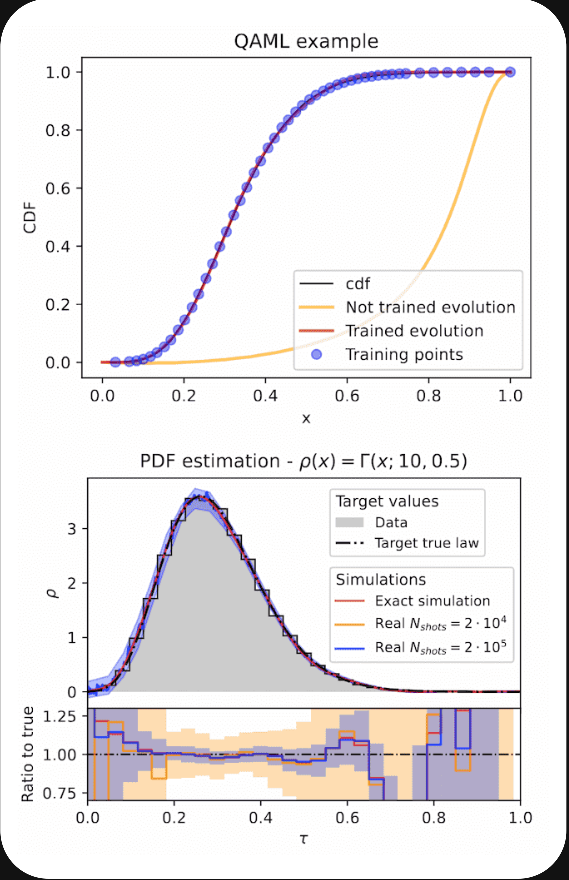
Density estimation with adiabatic QML [arXiv: 2303.11346]

Determining ***Probability Density Functions (PDF)***

by fitting the corresponding Cumulative Density Function (CDF) using an adiabatic QML ansatz.

Algorithm's summary:

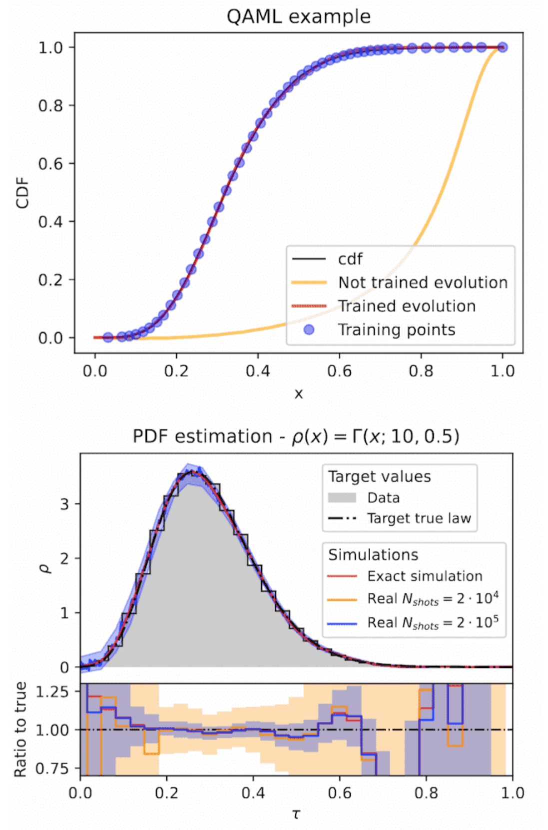
1. Optimize the parameters $\bar{\theta}$ using adiabatic evolution: $H_{ad}(\tau;\bar{\theta}) = [1 - s(\tau;\bar{\theta})]\hat{X} + s(\tau;\bar{\theta})\hat{Z}$ in order to approximate some target CDF values
2. Derivate from H_{ad} a circuit $\mathcal{C}(\tau;\bar{\theta})$ whose action on the ground state of $\hat{X}$ returns $|\psi(\tau)\rangle$
3. The circuit at step 2 can be used to calculate the CDF
4. Compute the PDF by derivating $\mathcal{C}$ with respect to τ using the Parameter Shift Rule



Density estimation with adiabatic QML [arXiv: 2303.11346]

Determining ***Probability Density Functions (PDF)***

by fitting the corresponding Cumulative Density Function (CDF) using an adiabatic QML ansatz.



Algorithm's summary:

1. Optimize the parameters $\bar{\theta}$ using adiabatic evolution: $H_{ad}(\tau;\bar{\theta}) = [1 - s(\tau;\bar{\theta})]\hat{X} + s(\tau;\bar{\theta})\hat{Z}$ in order to approximate some target CDF values
2. Derivate from H_{ad} a circuit $\mathcal{C}(\tau;\bar{\theta})$ whose action on the ground state of $\hat{X}$ returns $|\psi(\tau)\rangle$
3. The circuit at step 2 can be used to calculate the CDF
4. Compute the PDF by derivating $\mathcal{C}$ with respect to τ using the Parameter Shift Rule

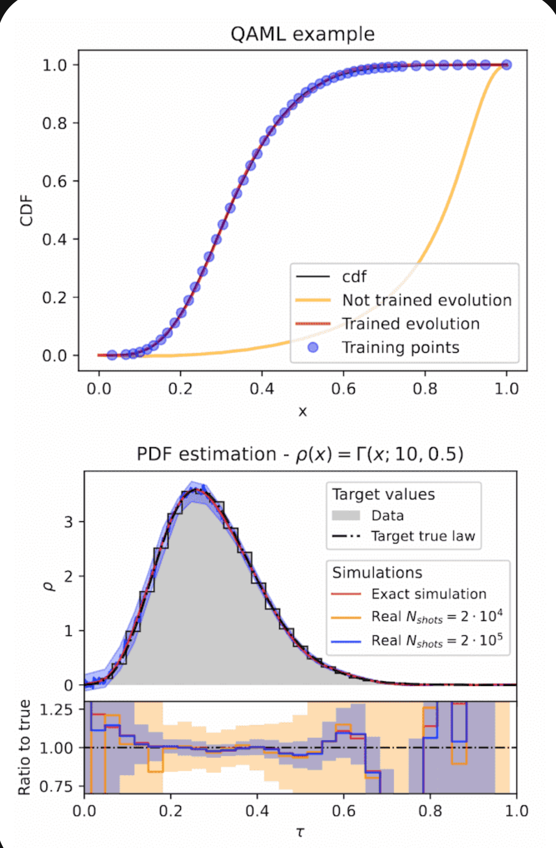
Density estimation with adiabatic QML [arXiv: 2303.11346]

Determining ***Probability Density Functions (PDF)***

by fitting the corresponding Cumulative Density Function (CDF) using an adiabatic QML ansatz.

Algorithm's summary

1. Optimize the parameters $\bar{\theta}$ using adiabatic evolution: $H_{ad}(\tau;\bar{\theta}) = [1 - s(\tau;\bar{\theta})]\hat{X} + s(\tau;\bar{\theta})\hat{Z}$ in order to approximate some target CDF values
2. Derivate from H_{ad} a circuit $\mathcal{C}(\tau;\bar{\theta})$ whose action on the ground state of $\hat{X}$ returns $|\psi(\tau)\rangle$
3. The circuit at step 2 can be used to calculate the CDF
4. Compute the PDF by derivating $\mathcal{C}$ with respect to τ using the Parameter Shift Rule



Quantum hardware

Quantum computation

Various models are proposed and explored

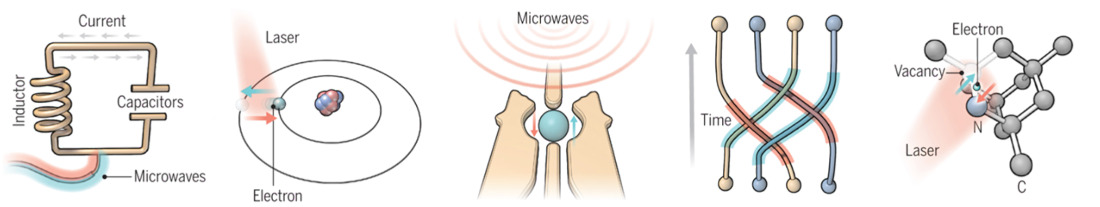
1. discrete gate-based
2. continuous variable (a.k.a. bosonic)
3. quantum annealing

The potential use cases partially overlap, and it is possible to emulate each other (at least approximately).

They are particularly related to the hardware realizing them...

Technologies

Many technologies simultaneously investigated `[arXiv: 2304.14360]`



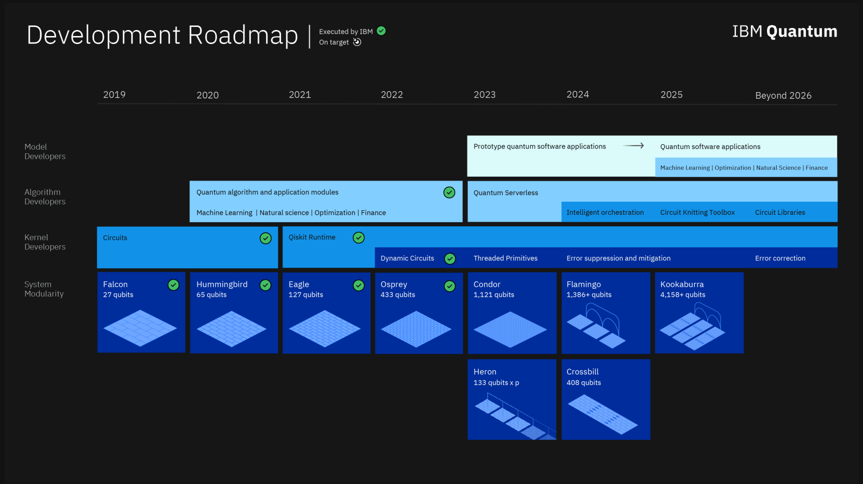
	Superconducting loops	Trapped ions	Silicon quantum dots	Topological qubits	Diamond vacancies
	A resistance-free current oscillates back and forth around a circuit loop. An injected microwave signal excites the current into super-position states.	Electrically charged atoms, or ions, have quantum energies that depend on the location of electrons. Tuned lasers cool and trap the ions, and put them in superposition states.	These "artificial atoms" are made by adding an electron to a small piece of pure silicon. Microwaves control the electron's quantum state.	Quasiparticles can be seen in the behavior of electrons channeled through semi-conductor structures. Their braided paths can encode quantum information.	A nitrogen atom and a vacancy add an electron to a diamond lattice. Its quantum spin state, along with those of nearby carbon nuclei, can be controlled with light.
Number entangled	9	14	2	N/A	6
Company support	Google, IBM, Quantum Circuits	ionQ	Intel	Microsoft, Bell Labs	Quantum Diamond Technologies
Pros	Fast working. Build on existing semiconductor industry.	Very stable. Highest achieved gate fidelities.	Stable. Build on existing semiconductor industry.	Greatly reduce errors.	Can operate at room temperature.
Cons	Collapse easily and must be kept cold.	Slow operation. Many lasers are needed.	Only a few entangled. Must be kept cold.	Existence not yet confirmed.	Difficult to entangle.

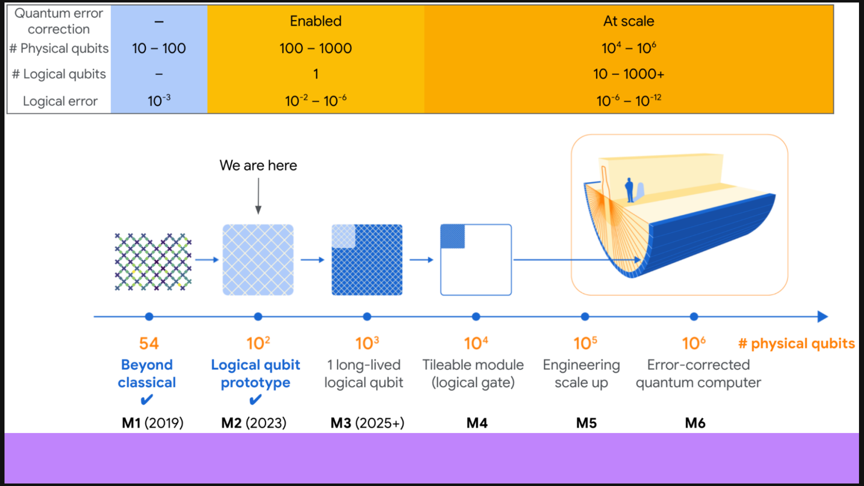
Pros and cons for each, investigated by different groups, including diverse private companies.

Some optimal for specific applications, others for further usage, e.g. quantum memories `[arXiv: 1511.04018]`.

Superconducting

One of the platforms with most resonance

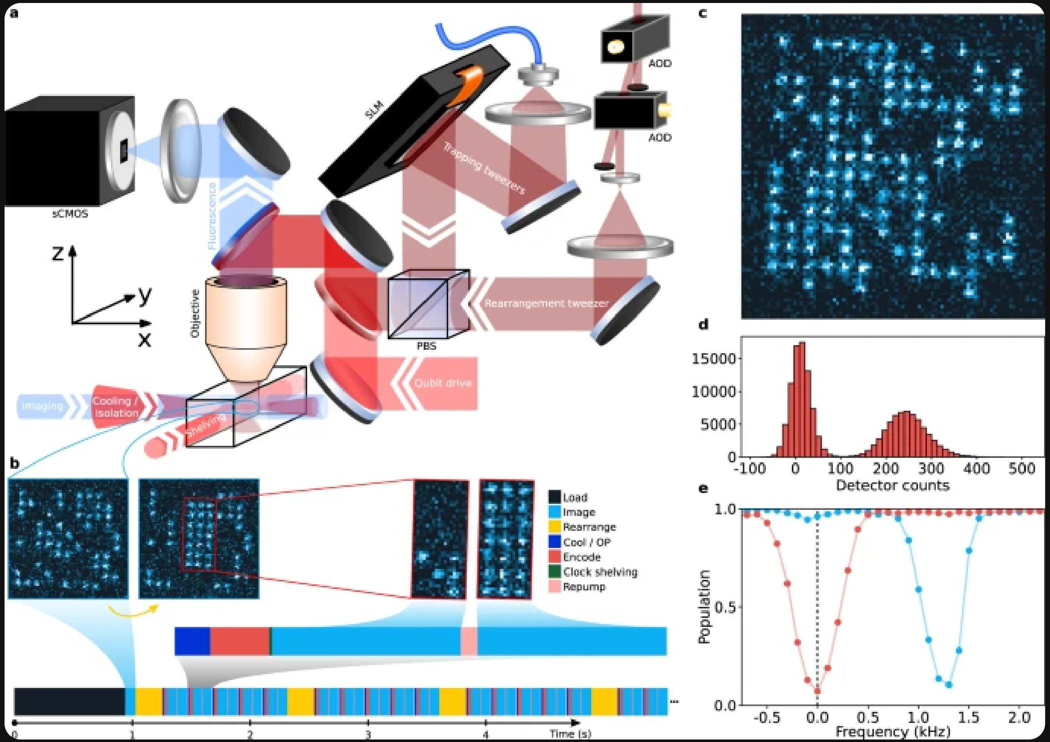




«IBM» and *«Google»* are definitely two prominent players, but superconducting hardware is being investigated by a plethora of labs.

Within the scope of this technology, many variations are also possible (flux-tunable qubits, couplers, cross-resonance schemes), so it is a macro-category.

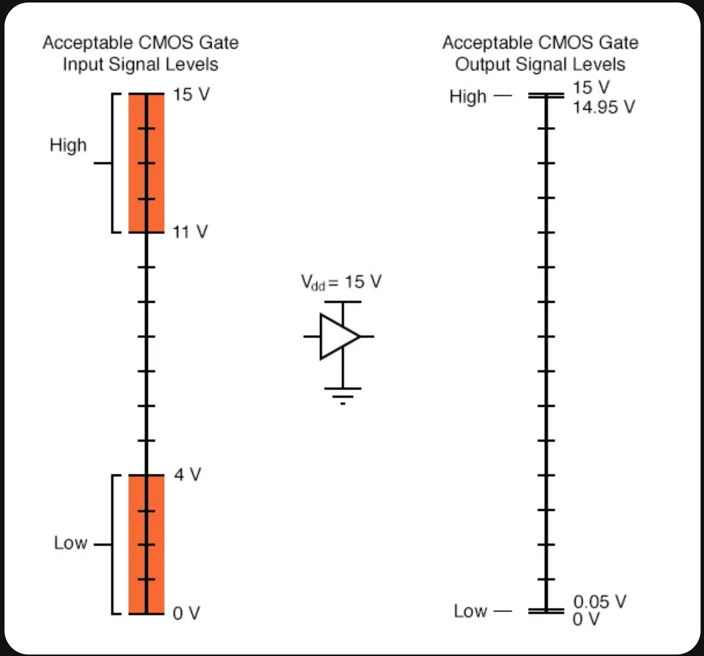
Neutral atoms

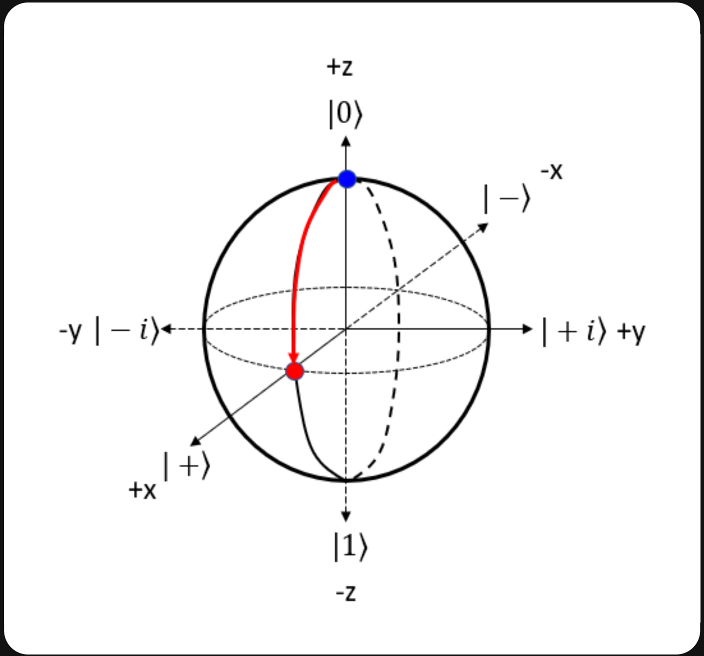


«Atom computing» have been the first to claim >1000 qubits `[arXiv: 2401.16177]`

Control

Quantum hardware is first of all an exercise in precise control



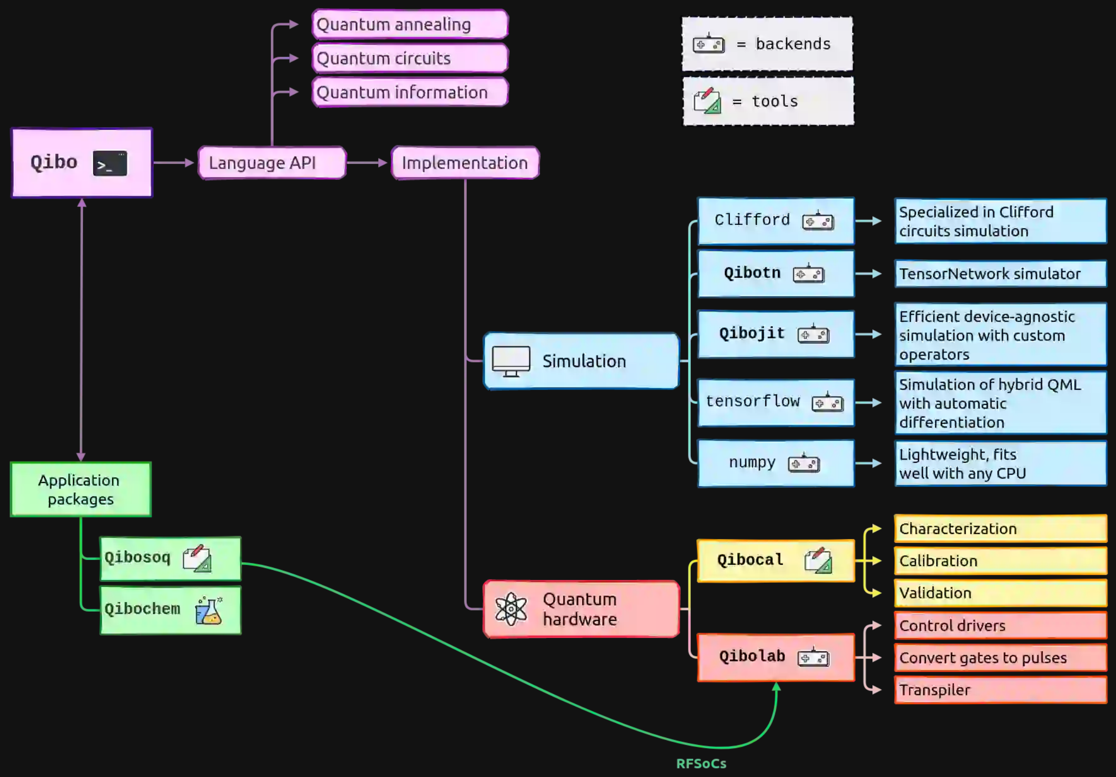


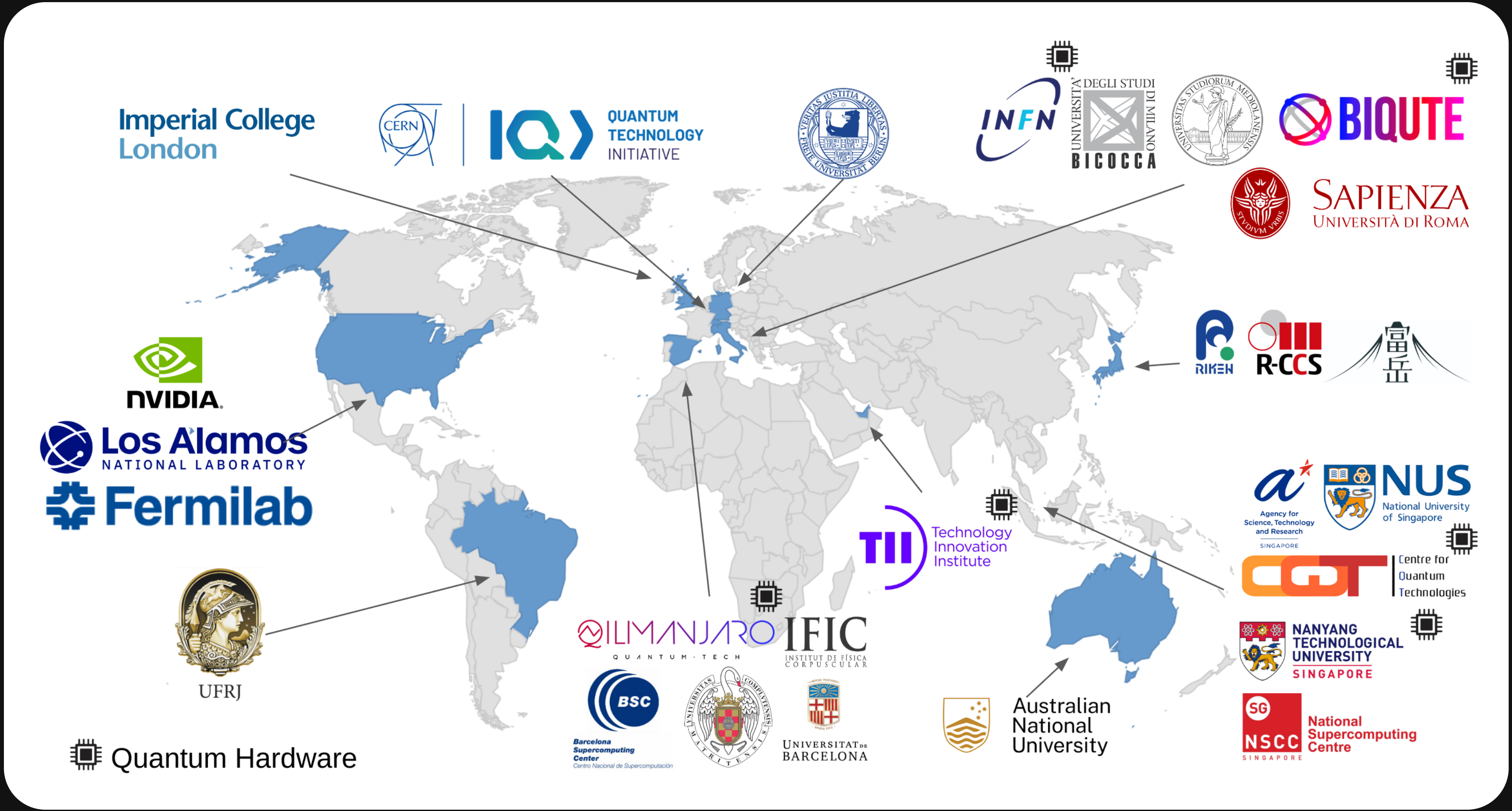
The quantum operation is supposed to be exact, not within a certain range.

Qibo

- Your quantum workhorse -

The ecosystem

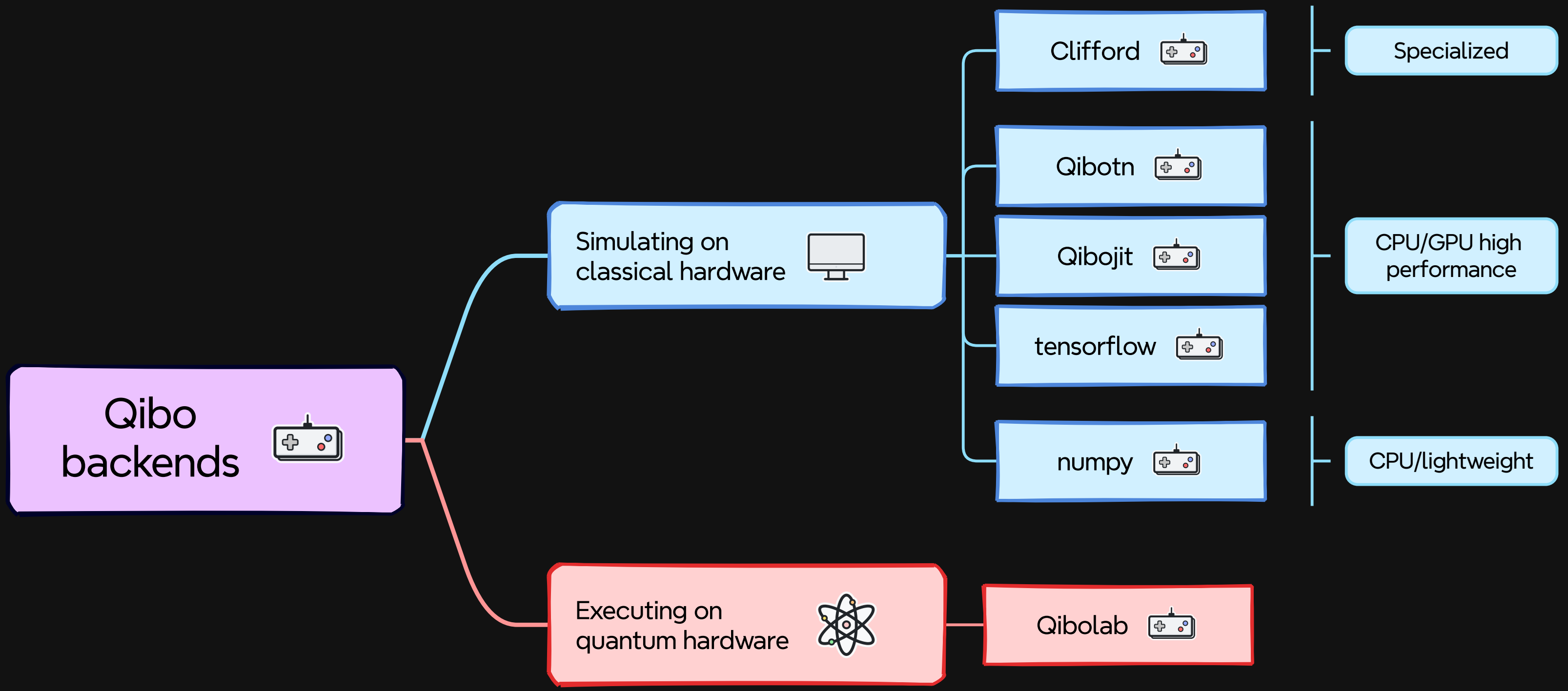




Contributors (March 2024)

Qibo [arXiv: 2009.01845]

Execution



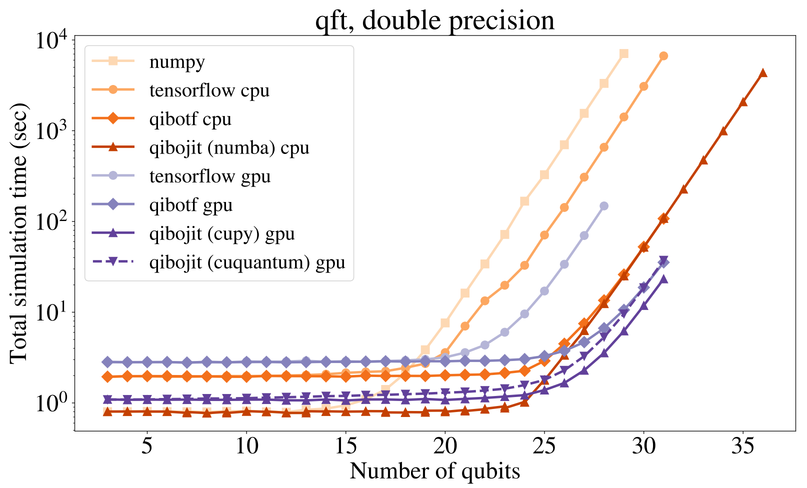
Backends mechanism

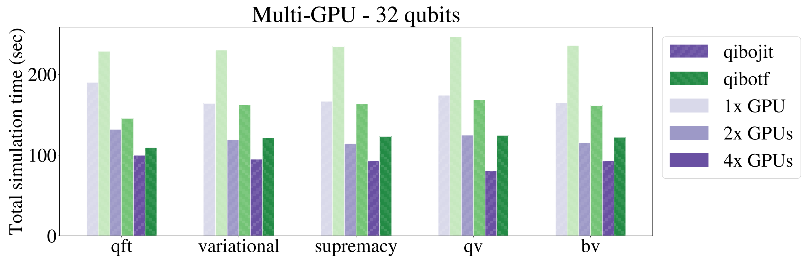
Plug the framework.

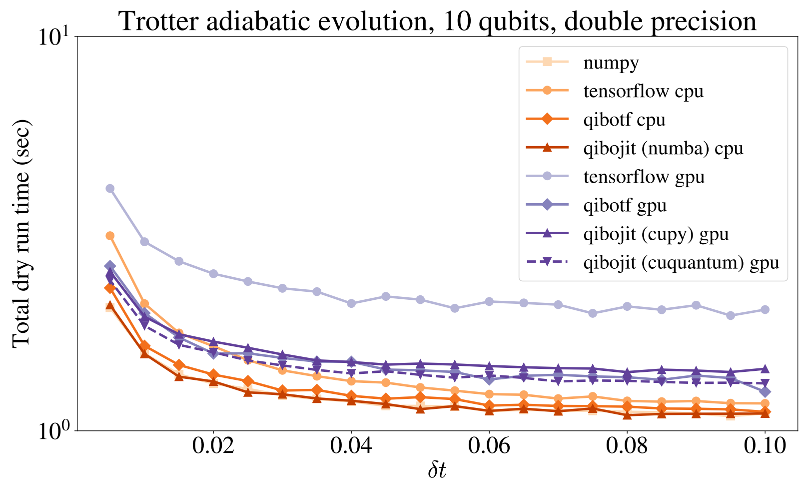
Structure the integration of the various libraries.

Common operations are implemented once and reused (when possible).

Results [arXiv: 2203.08826]







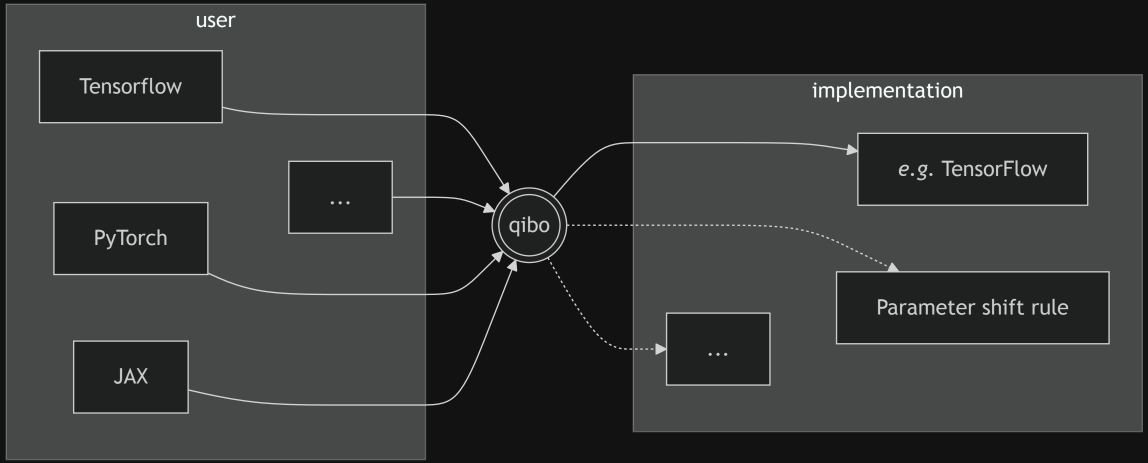
Automatic differentiation

for quantum machine learning → *Qiboml*

Autodiff simulation is fundamental to support QML investigation.

A dedicated differentiable backend in simulation can considerably help algorithms development.

Moving towards a single interface, encompassing both simulation and quantum hardware implementations.



Framework portability: implement in one, export derivatives.

Clifford

Specialized execution.

$$|\psi\rangle = U\,|\psi\rangle$$

Theorem 1 *Given an n-qubit state $|\psi\rangle$, the following are equivalent:*

(i) $|\psi\rangle$ can be obtained from $|0\rangle \otimes n$ by CNOT, Hadamard, and phase gates only.
(ii) $|\psi\rangle$ can be obtained from $|0\rangle \otimes n$ by CNOT, Hadamard, phase, and measurement gates only.
(iii) $|\psi\rangle$ is stabilized by exactly 2n Pauli operators.
*(iv) $|\psi\rangle$ **is uniquely determined by** $S(|\psi\rangle) = Stab(|\psi\rangle) \cap P_n$ **or the group of Pauli operators that stabilize** $|\psi\rangle$*

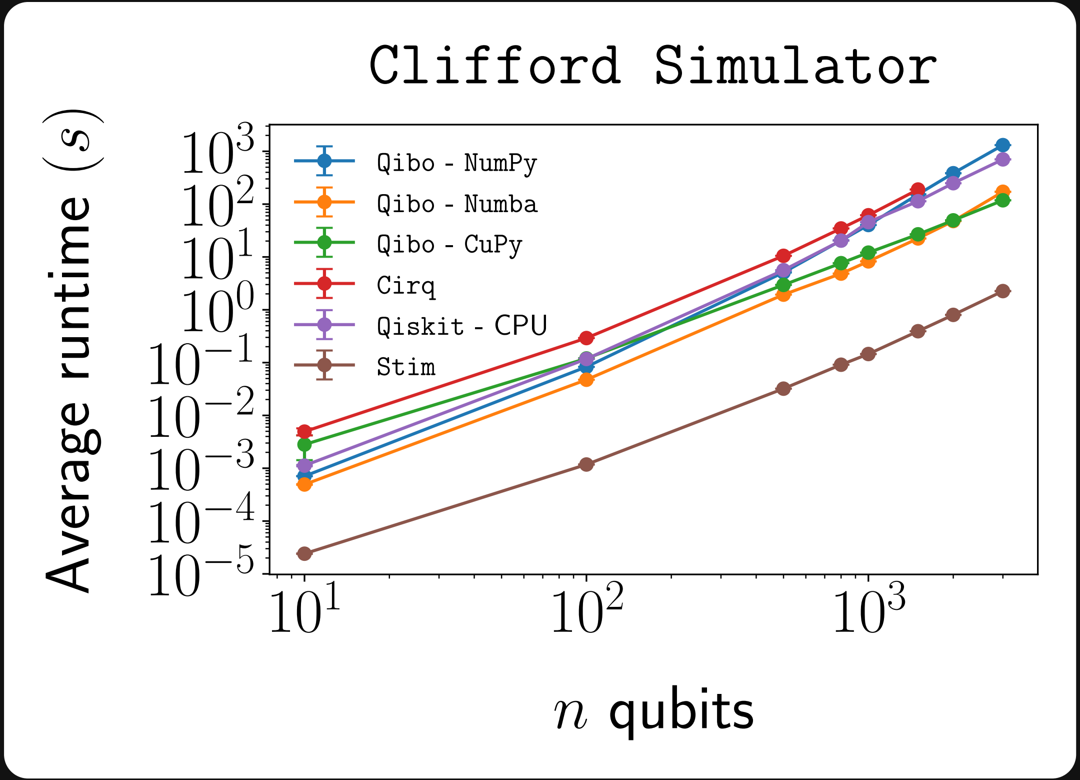
$$\left(\begin{array}{ccc|ccc|c} x_{11} & \cdots & x_{1n} & z_{11} & \cdots & z_{1n} & r_1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \cdots & x_{nn} & z_{n1} & \cdots & z_{nn} & r_n \\ \hline x_{(n+1)1} & \cdots & x_{(n+1)n} & z_{(n+1)1} & \cdots & z_{(n+1)n} & r_{n+1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{(2n)1} & \cdots & x_{(2n)n} & z_{(2n)1} & \cdots & z_{(2n)n} & r_{2n} \end{array}\right)$$

Instead of operating on the whole state vector, the state is represented by a much more compressed *tableau*.

It still requires vectorized operations on the boolean entries, that can be optimized in a similar fashion to the general state vector approach.

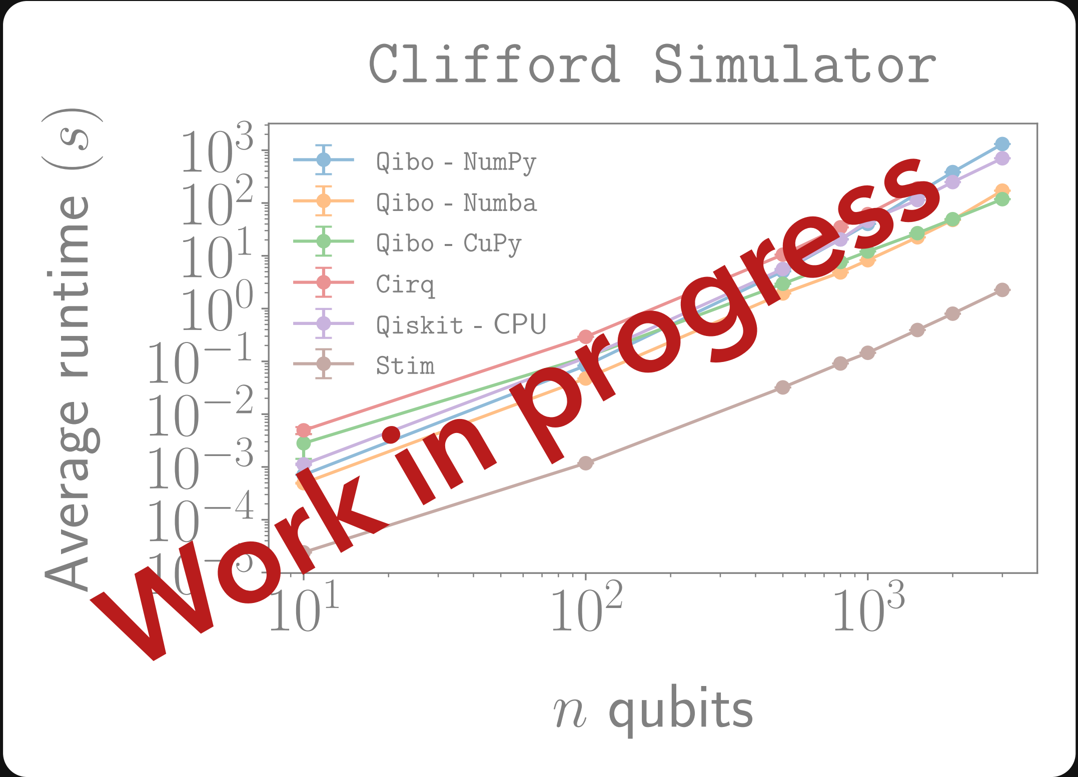
Clifford

Benchmarks



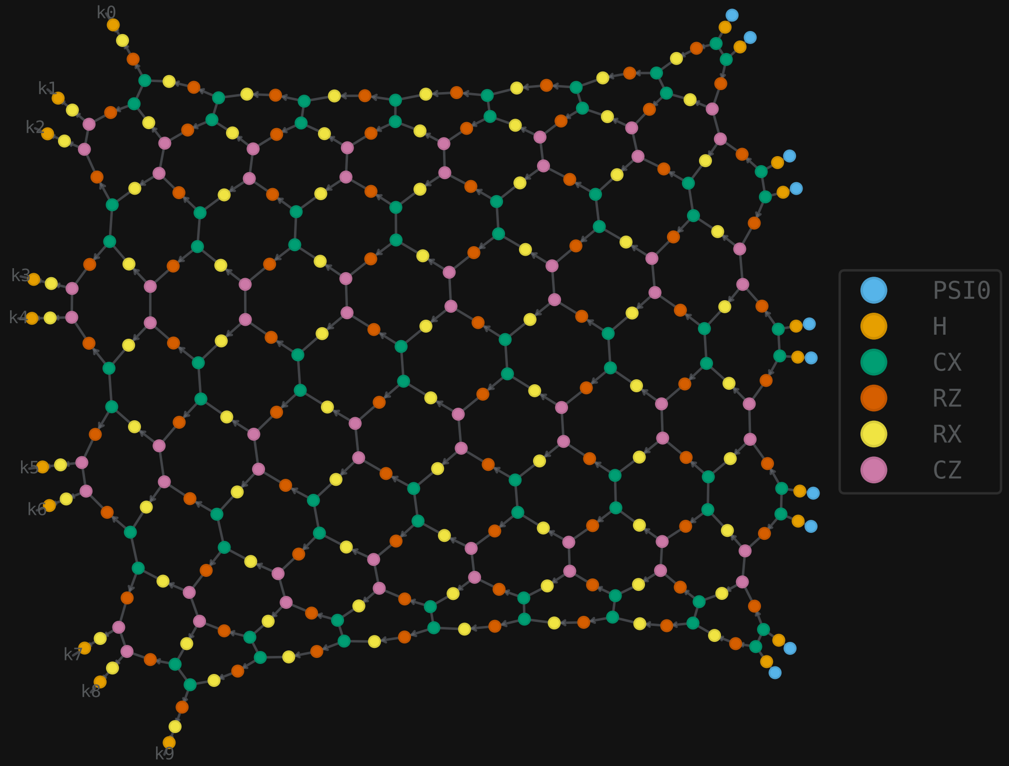
Clifford

Benchmarks

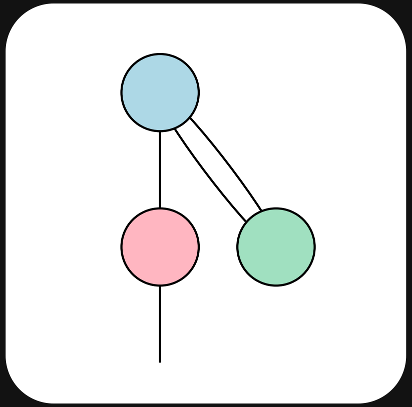


Tensor network

Optimized for observables.

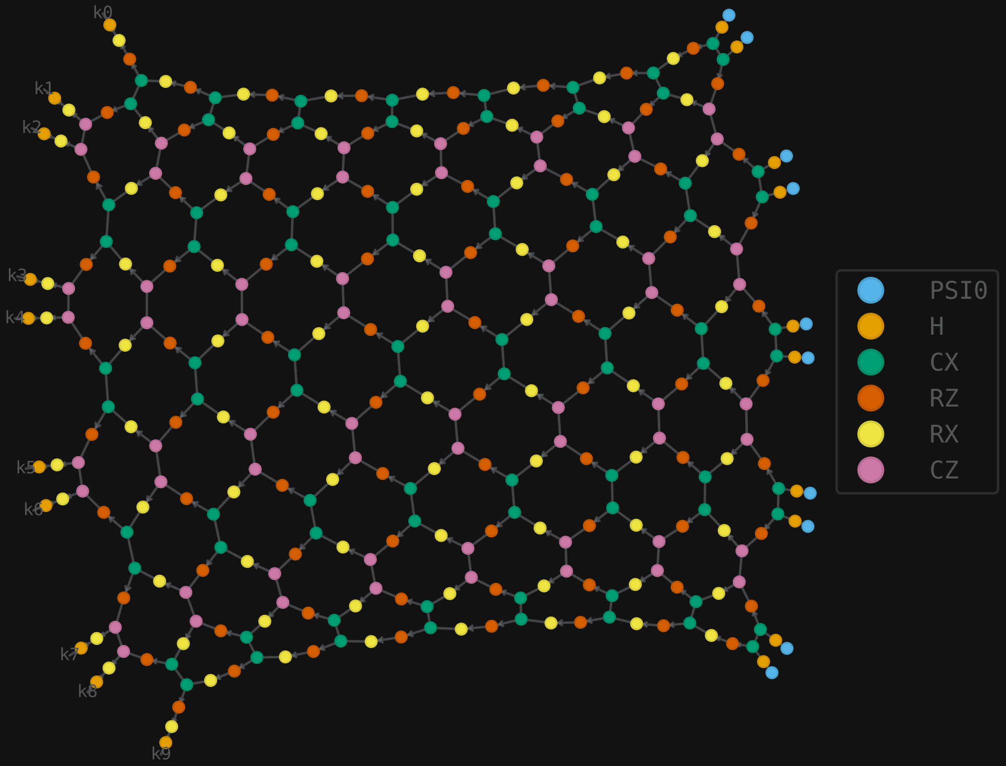


Contractions

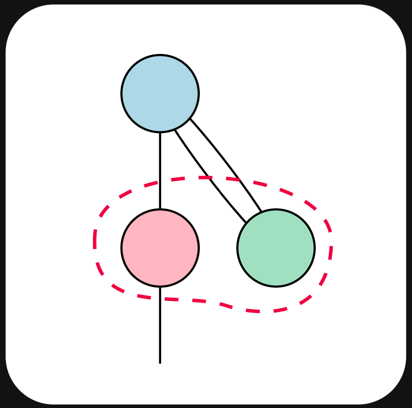


Tensor network

Optimized for observables.

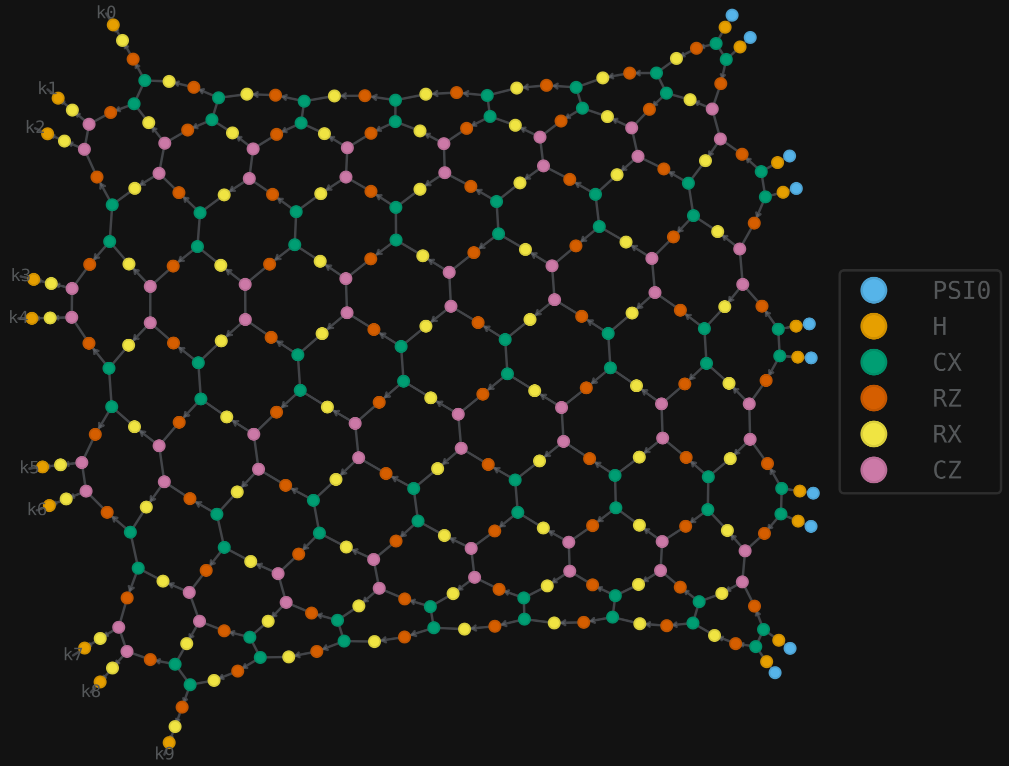


Contractions

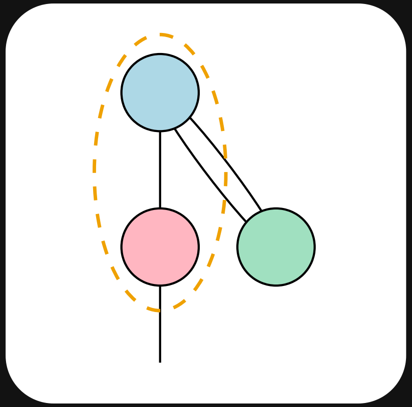


Tensor network

Optimized for observables.

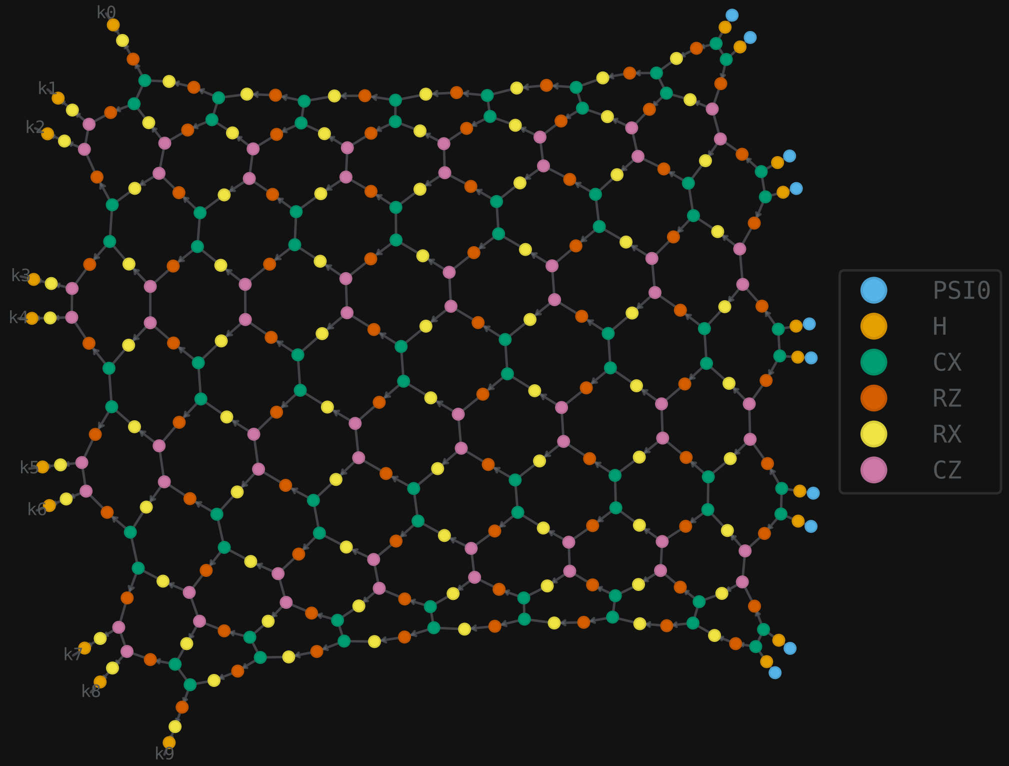


Contractions

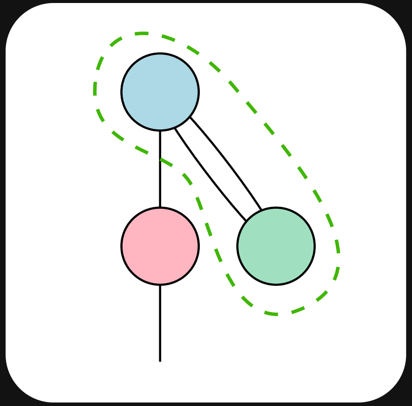


Tensor network

Optimized for observables.



Contractions

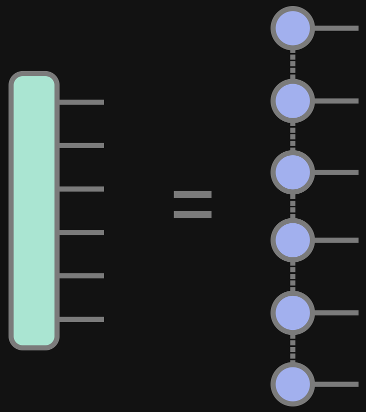


Tensor network

beyond `opt_einsum`

Approximation

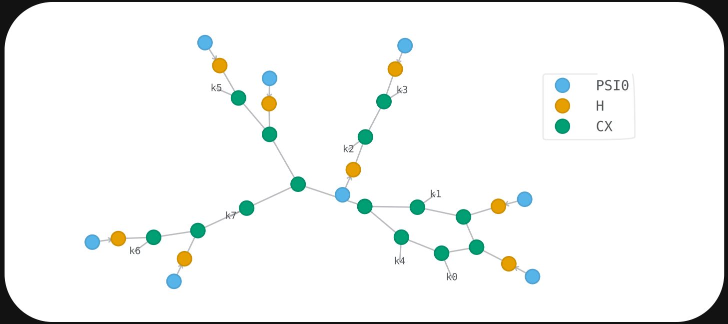
Based on singular value decomposition (SVD).



A very frequent matrix product state (MPS).

But also other ansatzes are used.

Workload distribution



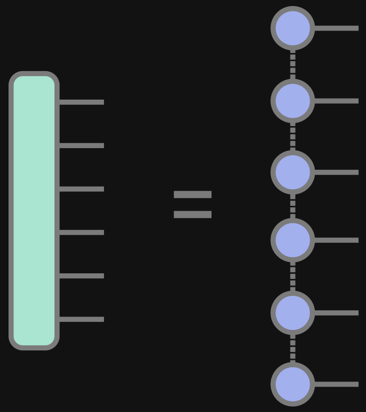

```python
for q in range(nq):
    c.apply_gate('H', q)

for q in range(0, nq, 2):
    c.apply_gate('CNOT', q, q + 1)

c.apply_gate('CNOT', 4, 7)
c.apply_gate('CNOT', 4, 1)
c.apply_gate('CNOT', 4, 0)
```

Tensor network

beyond `opt_einsum`

Approximation

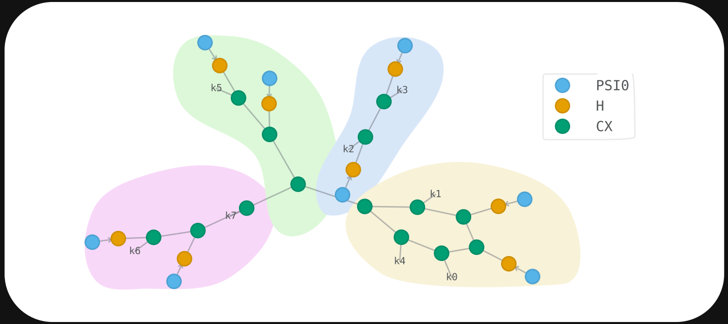
Based on singular value decomposition (SVD).



A very frequent matrix product state (MPS).

But also other ansatzes are used.

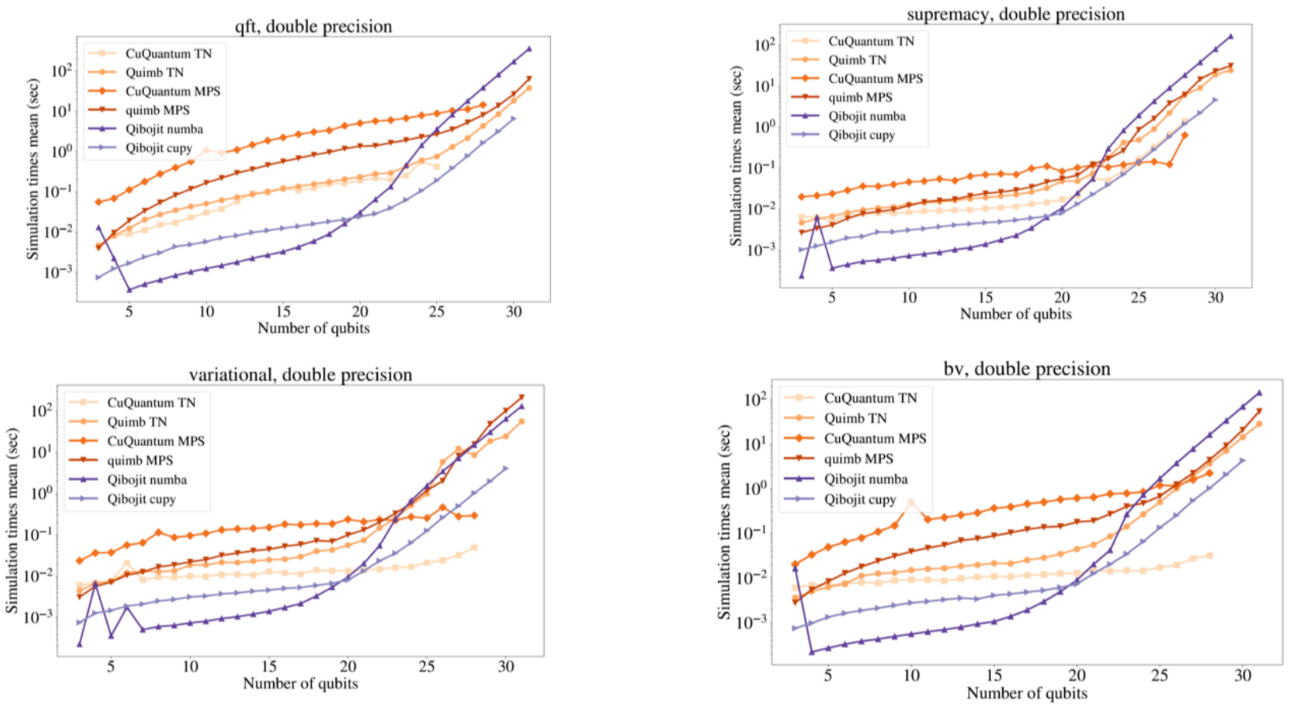
Workload distribution




```
for q in range(nq):
    c.apply_gate('H', q)

for q in range(0, nq, 2):
    c.apply_gate('CNOT', q, q + 1)

c.apply_gate('CNOT', 4, 7)
c.apply_gate('CNOT', 4, 1)
c.apply_gate('CNOT', 4, 0)
```

QiboTN

Benchmarking : Quantum states



QiboTN

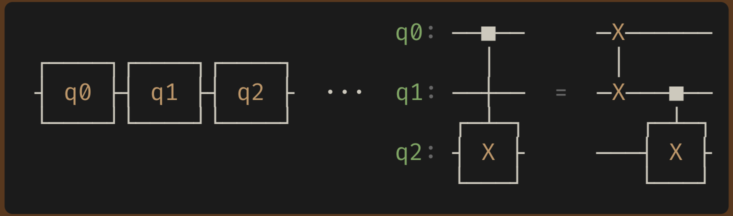
Benchmarking : Quantum states
qft, double precision
supremacy, double precision
variational, double precision
bv, double precision
CuQuantum TN
Quimb TN
CuQuantum MPS
quimb MPS
Qibojit numba
Qibojit cupy
Simulation times mean (sec)
Number of qubits
Work in progress

Transpilation

-- the bridge to hardware

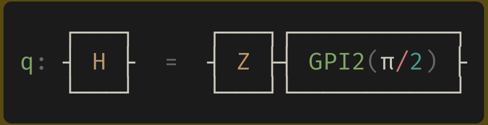
Optimization

Routing



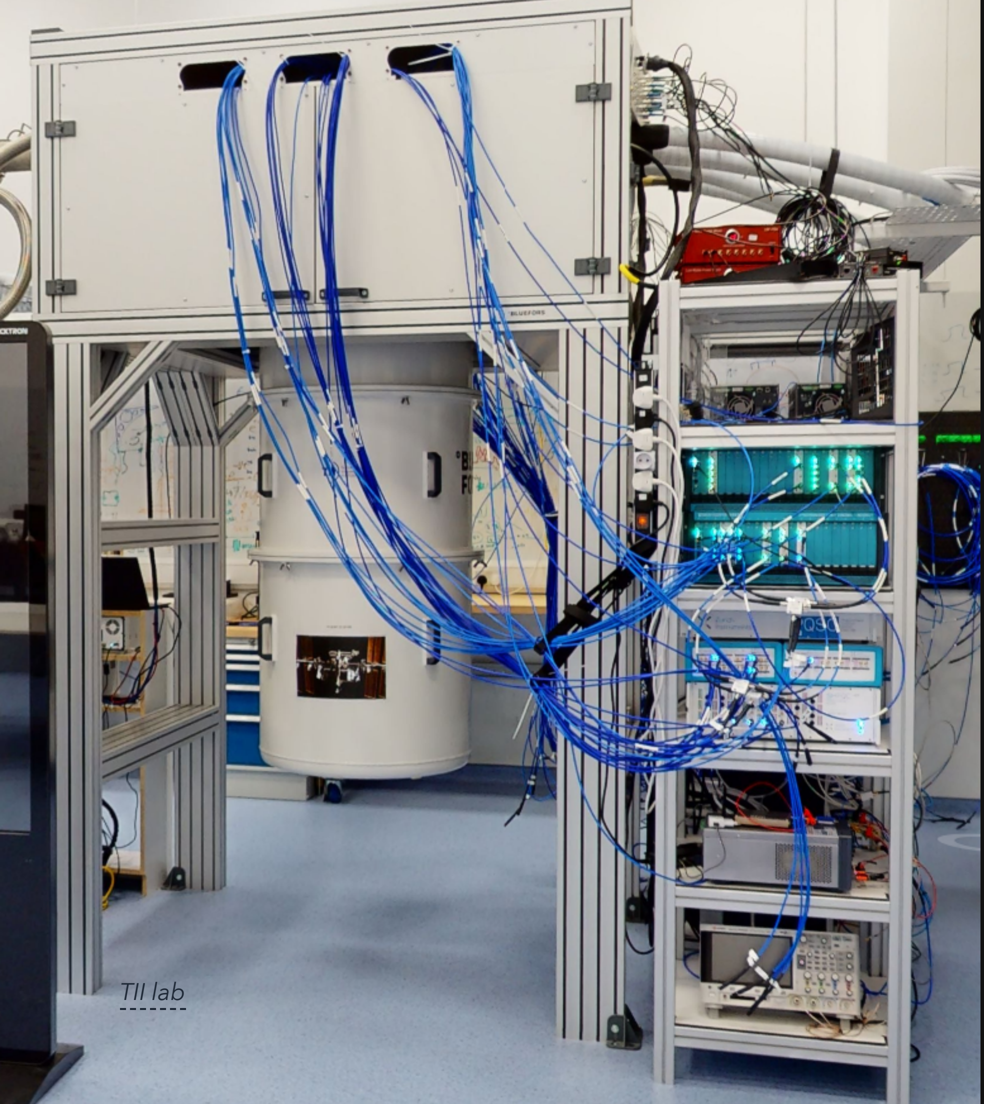
Decomposition (to natives)

Final assembly *lowering*.



→ COMPILATION

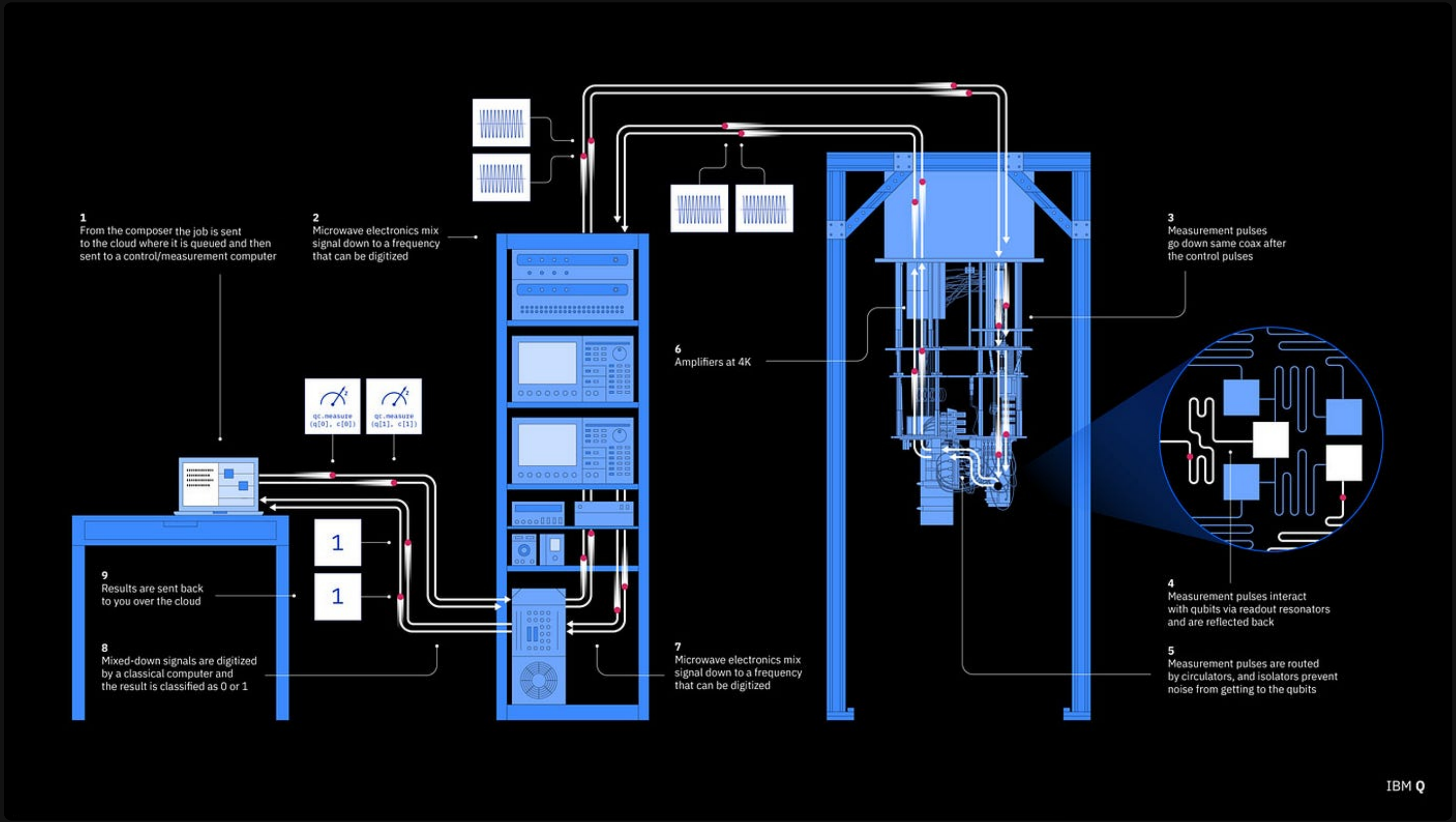
**simplicity is not well-defined, as in Mathematica and `gcc` → heuristics involved!*



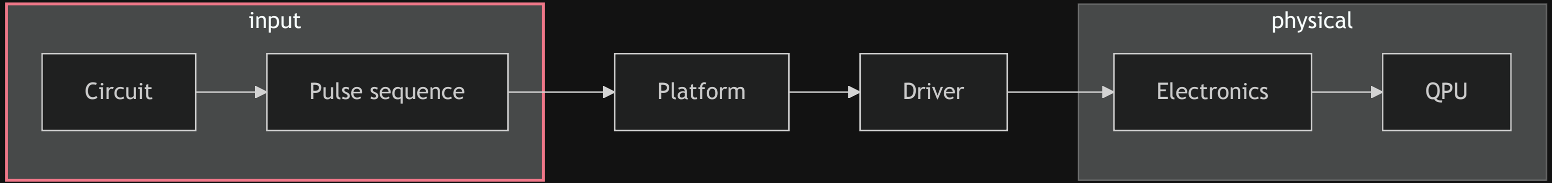
Qibolab `[arXiv: 2308.06313]`

Quantum control

Execution flow

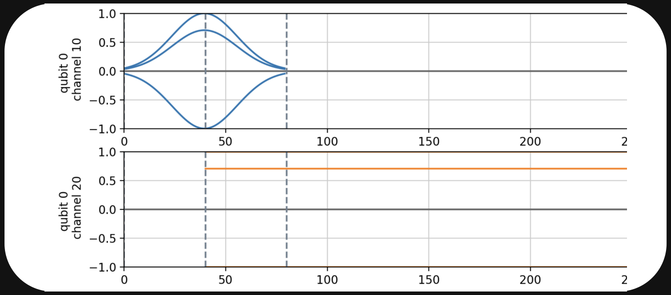


Qibolab - Interface



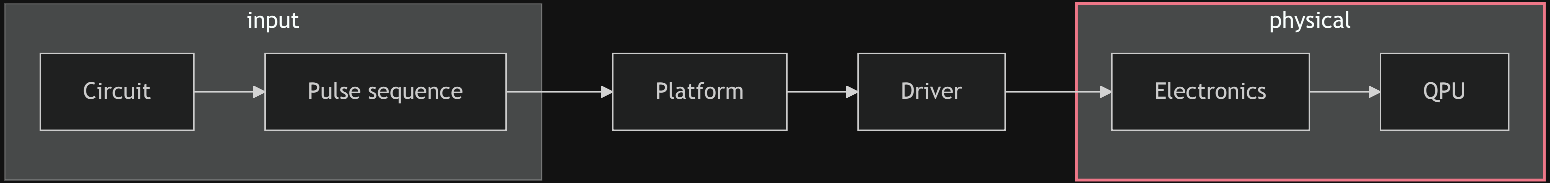
The **input** for a computation could be very standard, at the level of a **circuit**. That kind of interface is already defined by Qibo itself.

However, at a lower level, **pulses** are still a standard-enough way to interact with hardware, and these are defined by Qibolab.




```python
def create():
    instrument = DummyInstrument("myinstr", "0.0.0.0:0")

    channels = ChannelMap()
    channels |= Channel(
        "readout",
        port=instrument.ports("o1")
    )
    ...

    return Platform(
        "myplatform",
        qubits={qubit.name: qubit},
        instruments={instrument.name: instrument},
        ...
    )
```

Qibolab - Drivers

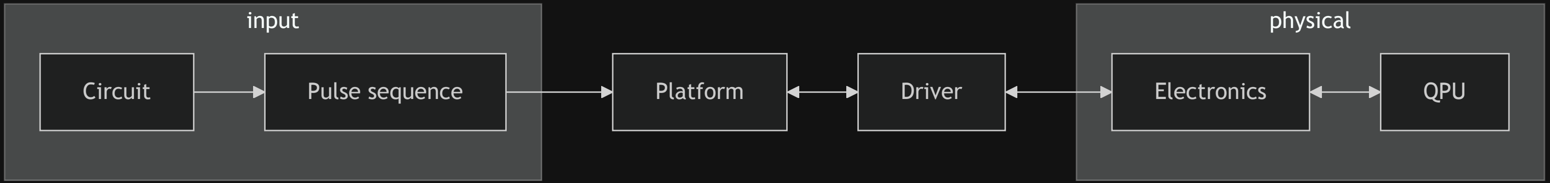


- Qblox
- Zurich
- QM
- QICK

```
       move      1,R0        # Start at marker output channel 0 (move 1 into R0)
       nop                   # Wait a cycle for R0 to be available.

loop:  set_mrk   R0          # Set marker output channels to R0
       upd_param 1000        # Update marker output channels and wait 1μs.
       asl       R0,1,R0     # Move to next marker output channel (left-shift R0).
       nop                   # Wait a cycle for R0 to be available.
       jlt       R0,16,@loop # Loop until all 4 marker output channels have been set once.

       set_mrk   0           # Reset marker output channels.
       upd_param 4           # Update marker output channels.
       stop                  # Stop sequencer.
```

by Qblox

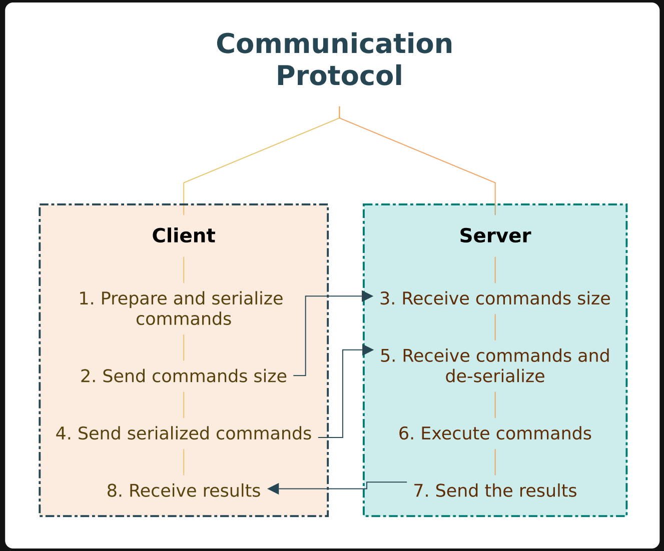
Qibosoq - Server on QICK `[arXiv: 2310.05851]`



Qibolab handles the whole connection, and takes care of fetching the single or multiple results.

For the single open source platform *FPGA FIRMWARE* currently in Qibolab, there has been a dedicate effort to define a suitable server, to optimize the communication with the board.

⟶ Qibosoq

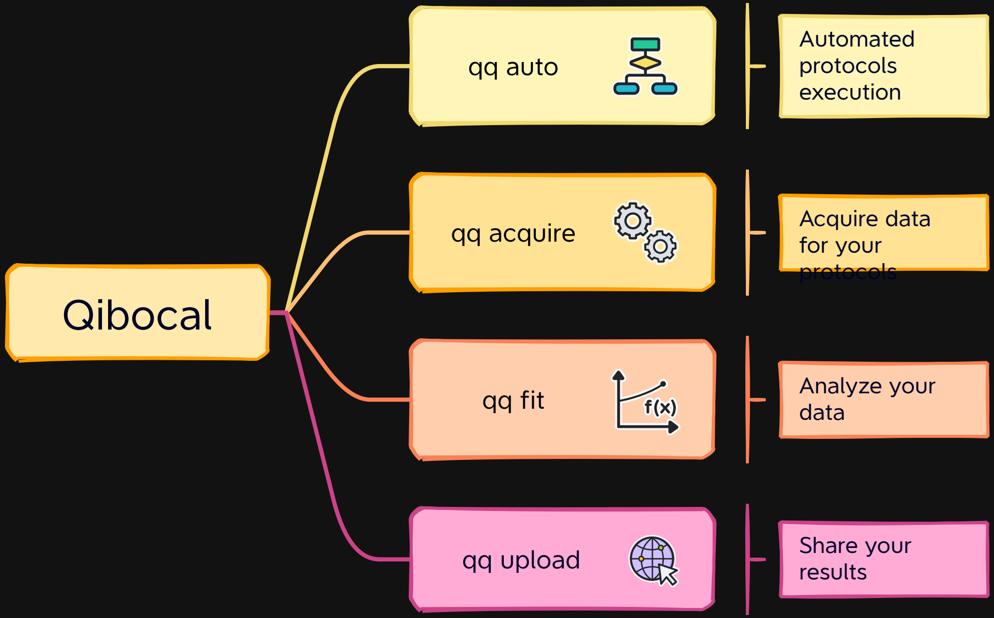


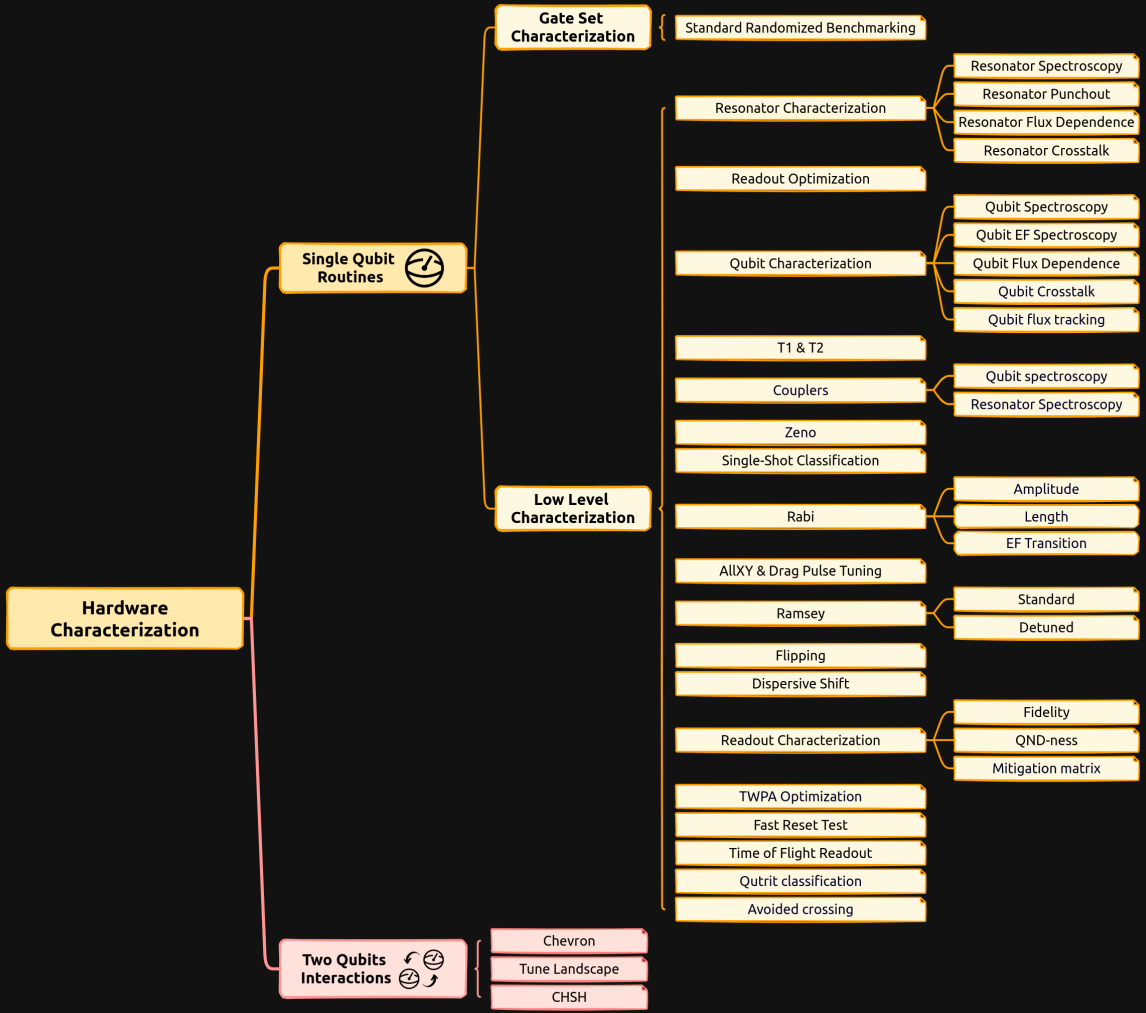
*in collaboration with **INFN-UNIMIB-BIQUTE***

Platform dashboard

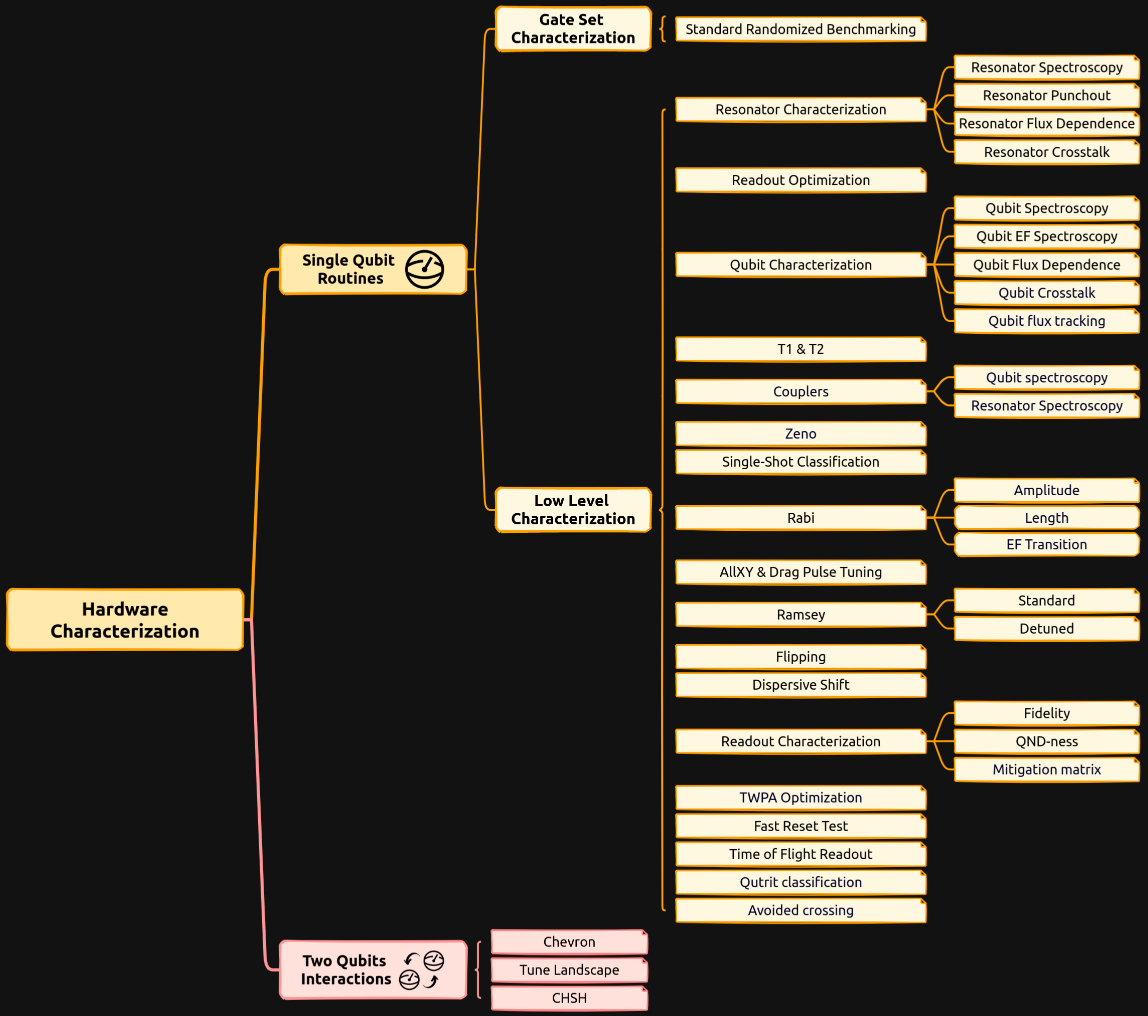
Qibocal `[arXiv: 2303.10397]`

A due mention





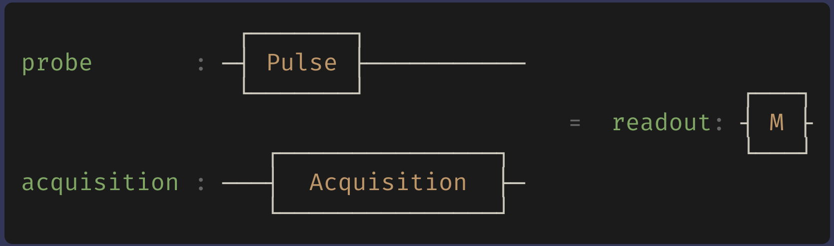
- characterize the hardware
- calibrate control
- validate performances



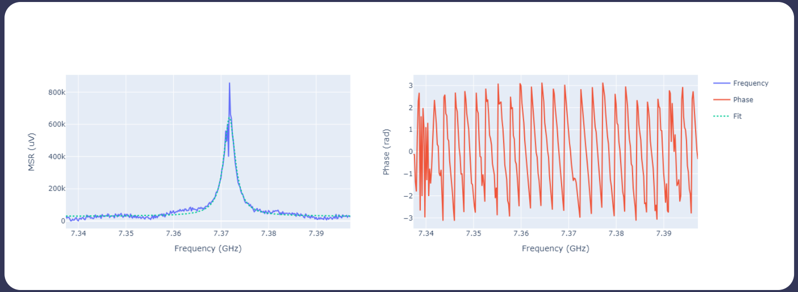
- characterize the hardware
- calibrate control
- validate performances

Pulses' calibration

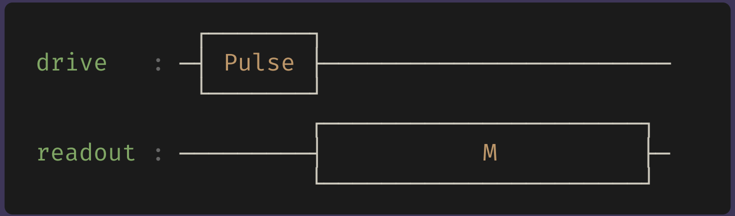
Resonator spectroscopy



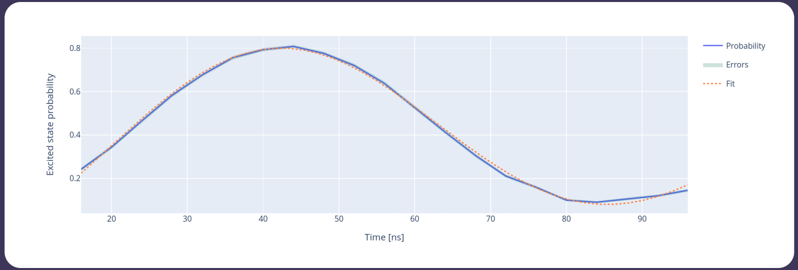
Scan spectrum to identify the coupled resonator frequency.



Rabi



Tune the amplitude (duration) of the drive pulse, in order to excite the qubit from the ground state up to state $|1\rangle$.



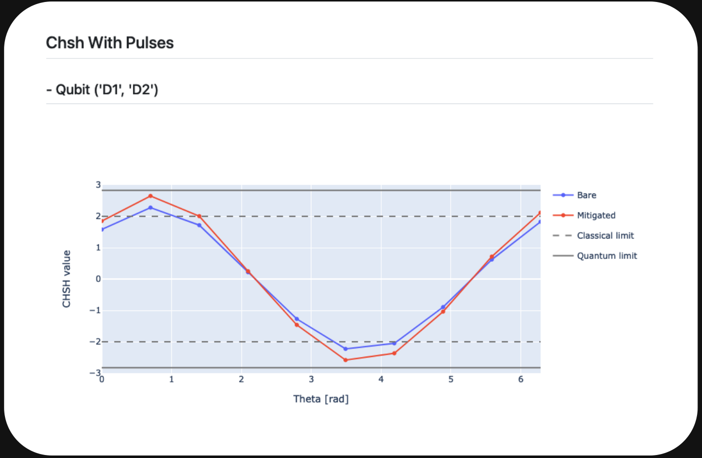
Protocols report

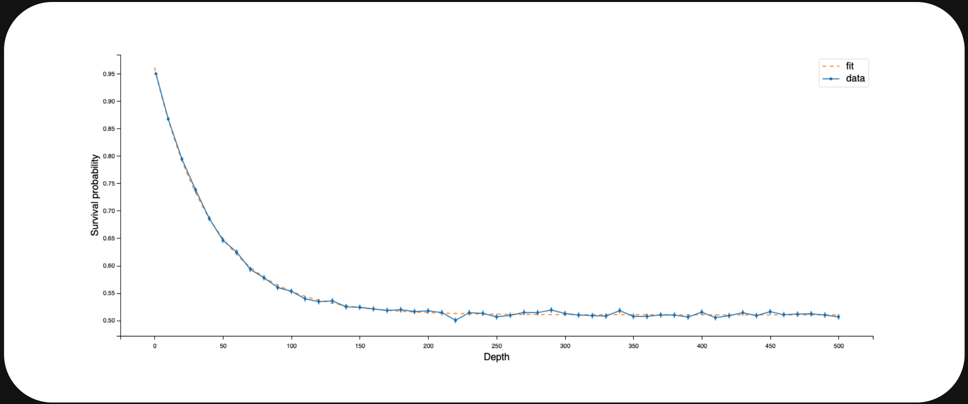
QPU control implementation

CHSH →

Randomized benchmarking ↓

They are two of the routines available in Qibocal, allowing to validate the QPU performances.





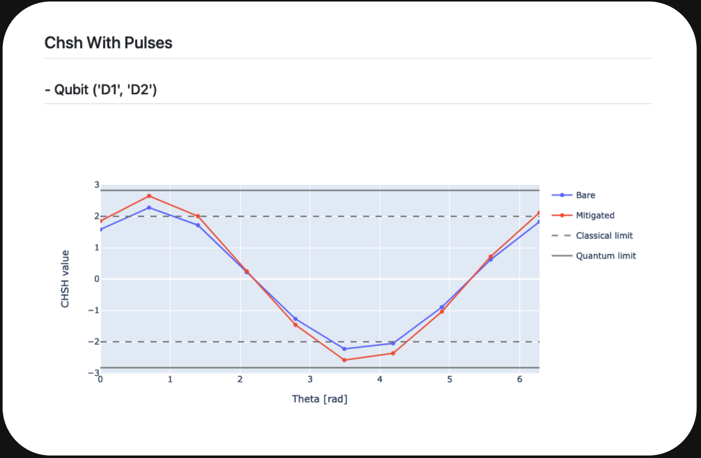
Protocols report

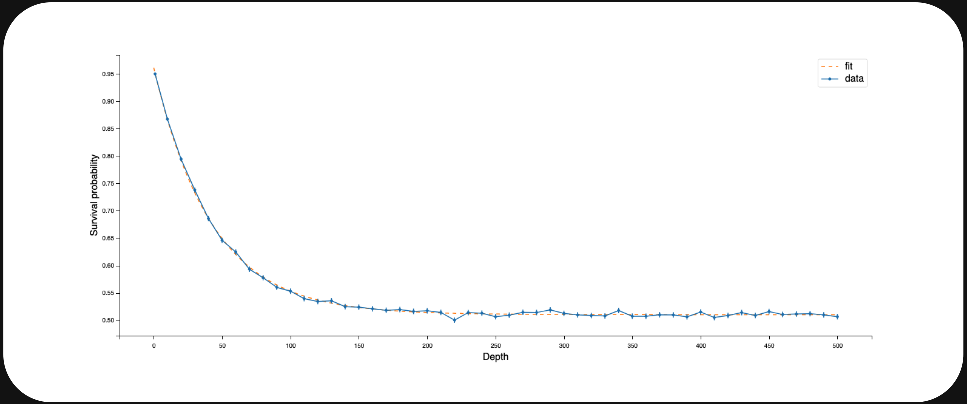
QPU control implementation

CHSH →

Randomized benchmarking ↓

They are two of the routines available in Qibocal, allowing to validate the QPU performances





Qibocal Reports

Uploaded Reports

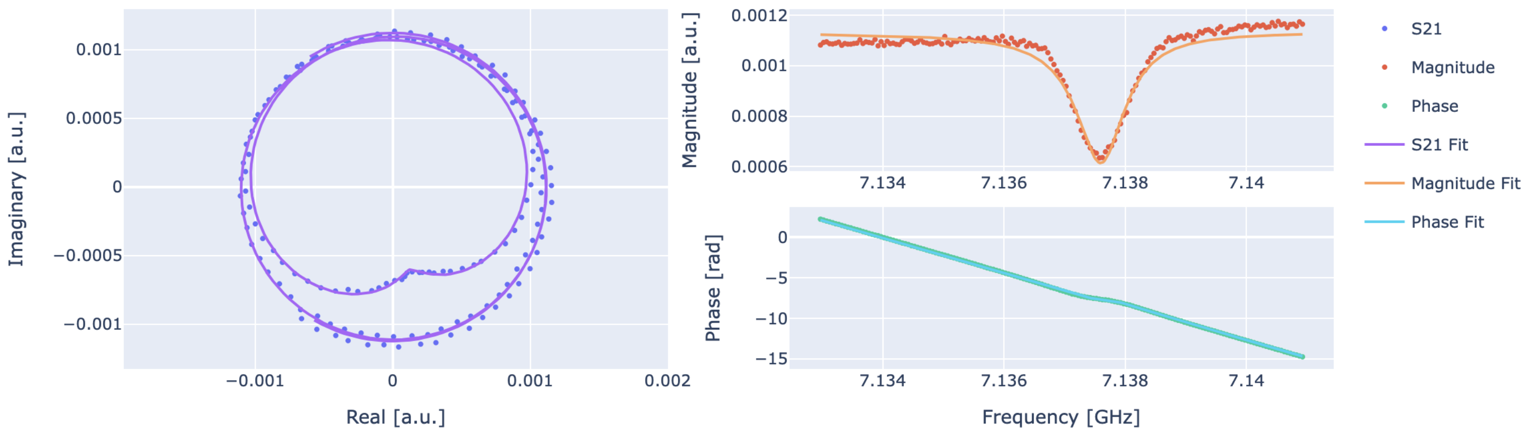
Please select a report from the table below:

Search:

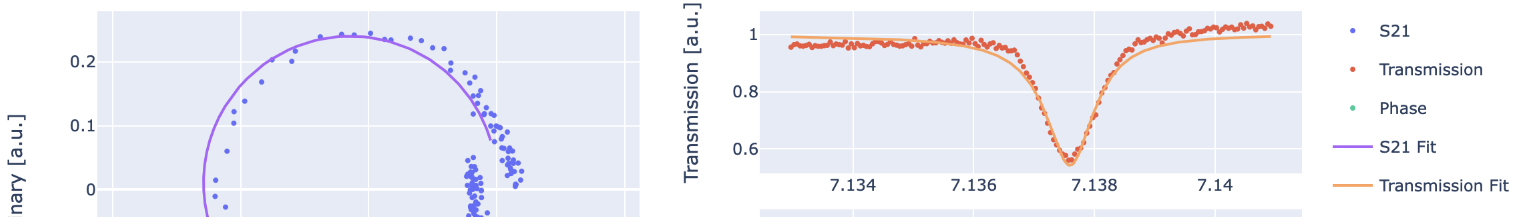
Title	Date	Platform	Start-time (UTC)	End-time (UTC)	Tag	Author
test_tii1qb1	2024-02-13	/home/users/andrea.pasquale/qibolab_platforms_qrc/tii1qs_xld1000	06:53:18	06:53:26	-	andrea.pasquale
test_qubit_spec_tii1qs	2024-02-13	/home/users/andrea.pasquale/qibolab_platforms_qrc/tii1qs_xld1000	06:59:45	06:59:50	-	andrea.pasquale
web_calibration_report_20240209_163420	2024-02-09	/home/users/qibocal/webapp/qibolab_platforms_qrc/iqm5q	12:34:25	12:34:51	web_calibration	qibocal
web_calibration_report_20240209_154537	2024-02-09	/home/users/qibocal/webapp/qibolab_platforms_qrc/iqm5q	11:45:54	11:46:21	web_calibration	qibocal
web_calibration_report_20240209_163420	2024-02-09	/home/users/qibocal/webapp/qibolab_platforms_qrc/iqm5q	12:34:25	12:34:51	web_calibration	qibocal

D1	Phase Shift [rad]	-5.788124e-01
D1	Electronic Delay [s]	3.390674e-07

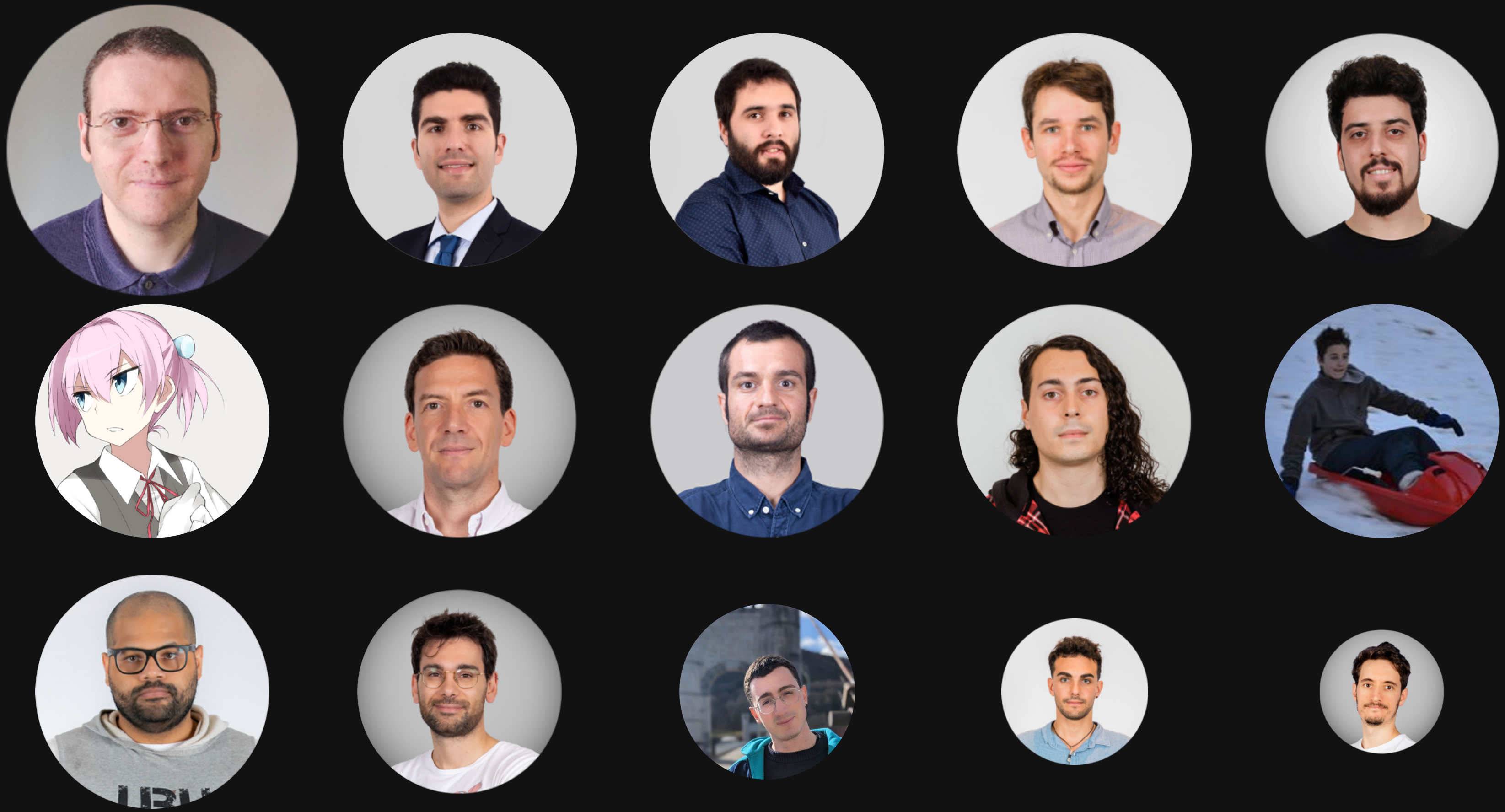
Raw data



Calibrated data



⩾ Not a one-man show...



Thanks