

TOWARDS A HYBRID QUANTUM OPERATING SYSTEM

Andrea Pasquale on the behalf of the Qibo collaboration

9th May 2023, CHEP2023, Norfolk



WHY QUANTUM COMPUTING IN HEP?

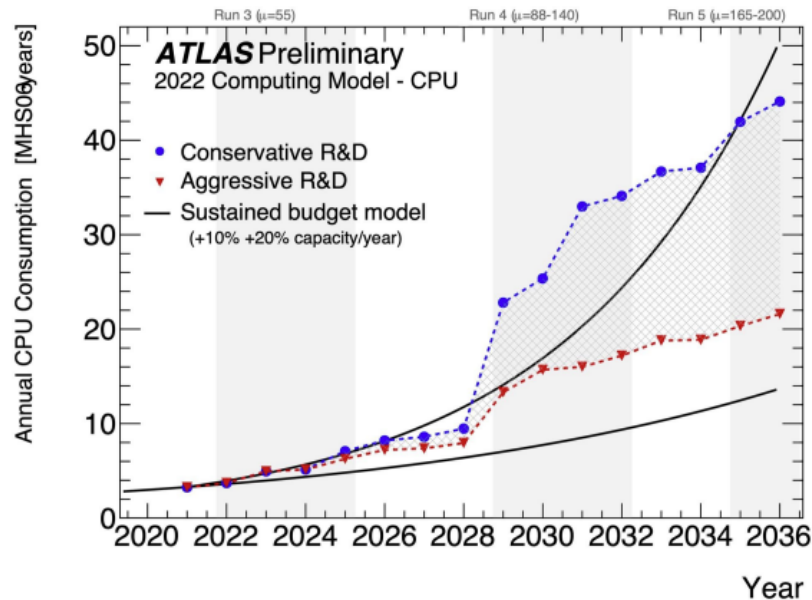
Nat Rev Phys 4, 143–144

LHC produces TB of data per second which need to be processed.

Currently we **cannot** keep up with the computational resources required!

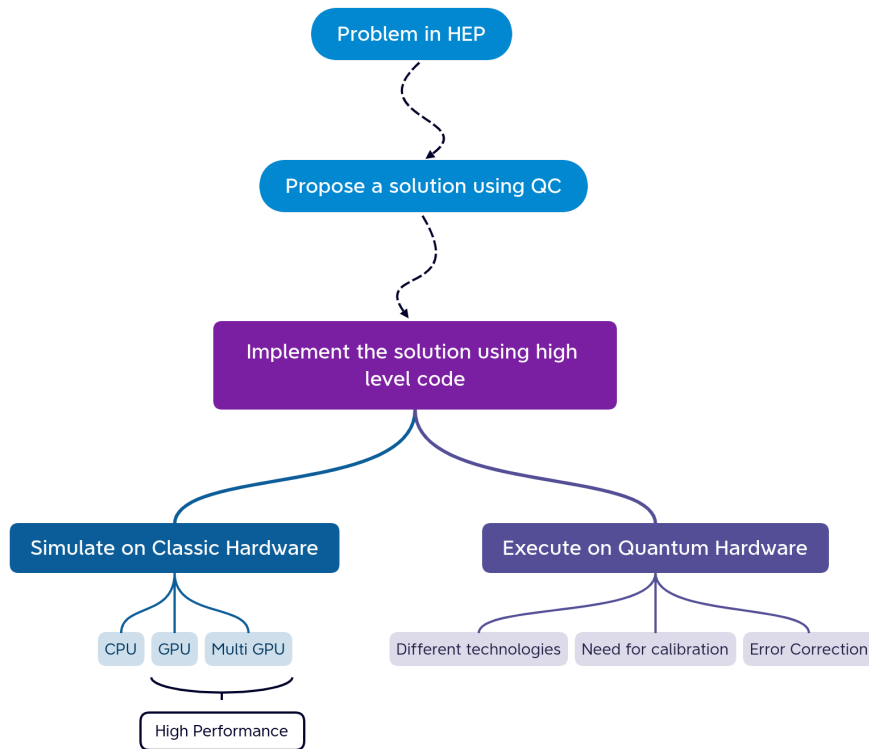
Quantum Computing (QC), especially Quantum Machine Learning (QML), is a promising solution:

- Efficient in **high-dimensional** quantum state spaces
- Potential computational speed-ups
- Qubit representation offers **better compression**
- Acceleration of **classically expensive** operations (eigensolvers).



Projected compute usage from ATLAS Software and Computing HC-LHC Roadmap

HOW CAN WE START USING QUANTUM COMPUTING?

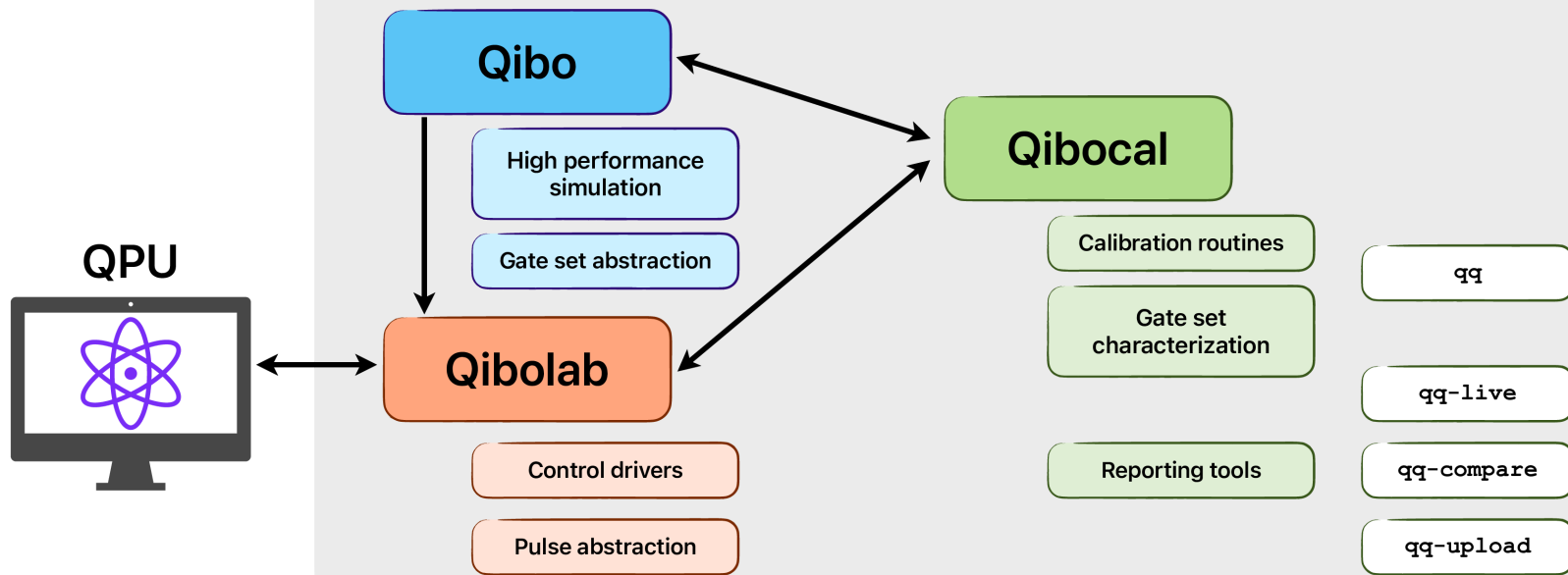


Is it possible to create from scratch a framework for all of this?

INTRODUCING QIBO

Open-source full stack API for quantum simulation, hardware control and calibration

Qibo framework



SIMULATION

GATE SET ABSTRACTION

```
import numpy as np
from qibo.models import Circuit
from qibo import gates, set_backend

# Set driver engine
set_backend("numpy")

c = Circuit(2)
c.add(gates.X(0))

# Add a measurement register on both qubits
c.add(gates.M(0, 1))

# Execute the circuit with the default initial state |00>.
result = c(nshots=100)

# Change backend
set_backend("qibojit")

# Circuit execution with new driver
result = c(nshots=100)
```

QIBO FEATURES

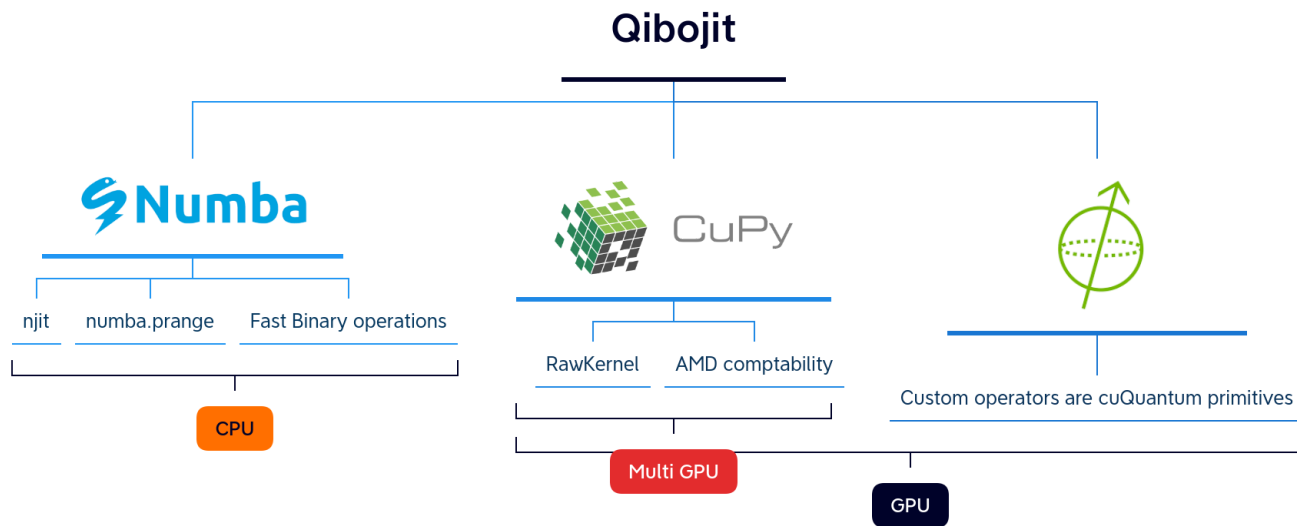
- Definition of a **standard language** for the construction and execution of quantum circuits with **device agnostic approach** to simulation and quantum hardware control based on plug and play backend drivers.
- A **continuously growing** code-base of quantum algorithms and applications presented with examples and tutorials.
- **Efficient simulation** backends with GPU, multi-GPU and CPU with multi-threading support.
- A simple mechanism for adding **new simulation and hardware backend drivers**.

2009.01845

HIGH PERFORMANCE SIMULATION

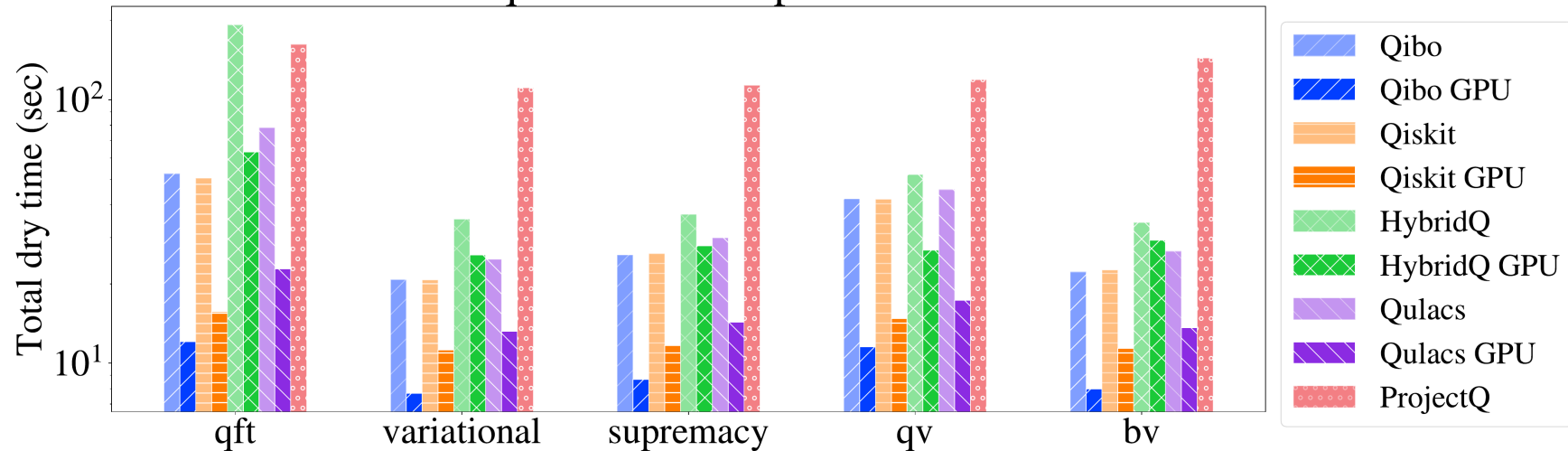
✗ Long computational times using naive approach (Numpy or TensorFlow) for circuits with large number of qubits.

✓ We need more sophisticated tools to be able to simulate a quantum circuits with more qubits!



BENCHMARK

30 qubits - double precision



All the benchmarks are available in [qibojit-benchmarks](#).



Automatic deployment of quantum circuits on quantum hardware

MOTIVATION

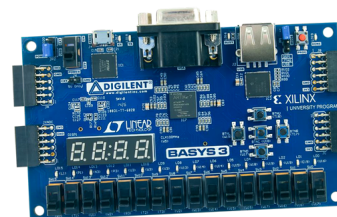
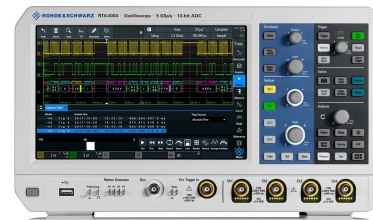
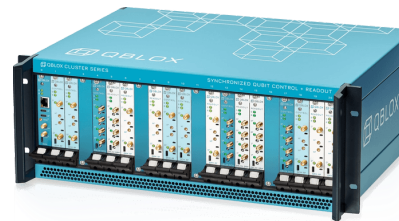
How are gates implemented on a quantum computer?

By sending microwave pulses.

How do we control them?

Using FPGAs

→ We need both a pulse level API + drivers to interface Qibo with different instruments.



PULSE API EXAMPLE

```
from qibolab import Platform
from qibolab.pulses import ReadoutPulse, PulseSequence

# Define PulseSequence
sequence = PulseSequence()
# Add some pulses to the pulse sequence
sequence.add(ReadoutPulse(start=0,
                           amplitude=0.3,
                           duration=4000,
                           frequency=200_000_000,
                           shape='Gaussian(5)'))

# Define platform
platform = Platform("tii1q_b1")
# Platform setup
platform.connect()
platform.setup()
platform.start()
# Executes a pulse sequence.
results = platform.execute_pulse_sequence()
platform.stop()
platform.disconnect()
```

DRIVERS IMPLEMENTED

Currently Qibolab supports the following drivers:

- Qblox
- Quantum Machines
- Zurich Instruments
- FPGAs (based on Qick)

We also support local oscillators

- RohdeSchwarz SGS100A
- ERASynth

INTRODUCING QIBOCAL

A reporting tool for calibration using Qibo

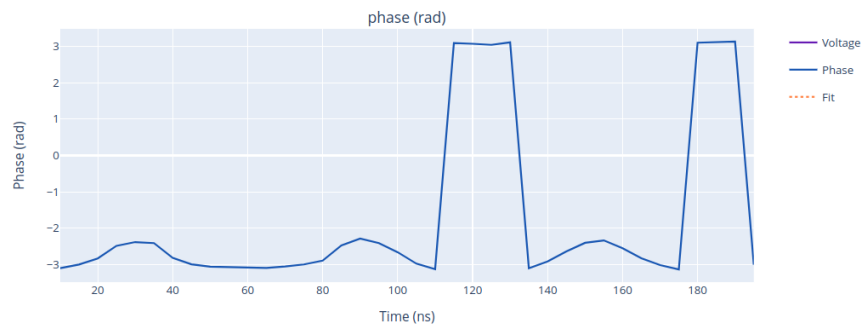
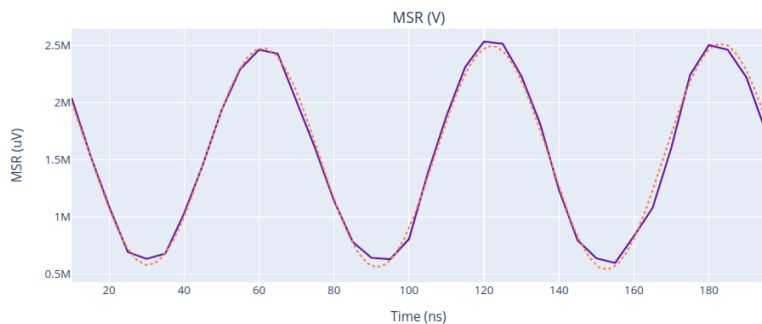
MOTIVATION

Let's suppose the following:

1. We have a QPU (self-hosted).
2. We have control over what we send to the QPU.
3. We know how to convert quantum circuits to pulses.

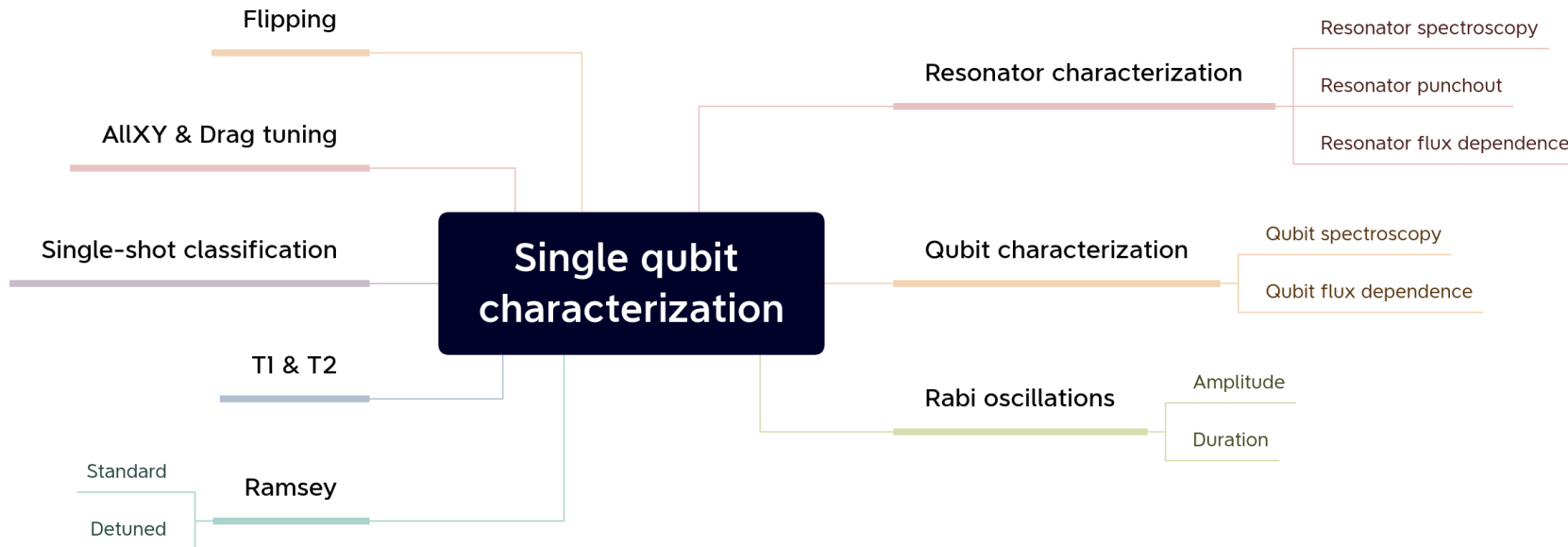
Can I **trust** my results? **NO!**

Characterization and **calibration** are an essential step to properly operate emerging quantum devices.



Calibration of RX pulse amplitude through a Rabi experiment through Qibocal.

SINGLE QUBIT CHARACTERIZATION: PULSE LEVEL



SINGLE QUBIT CHARACTERIZATION: CIRCUIT LEVEL

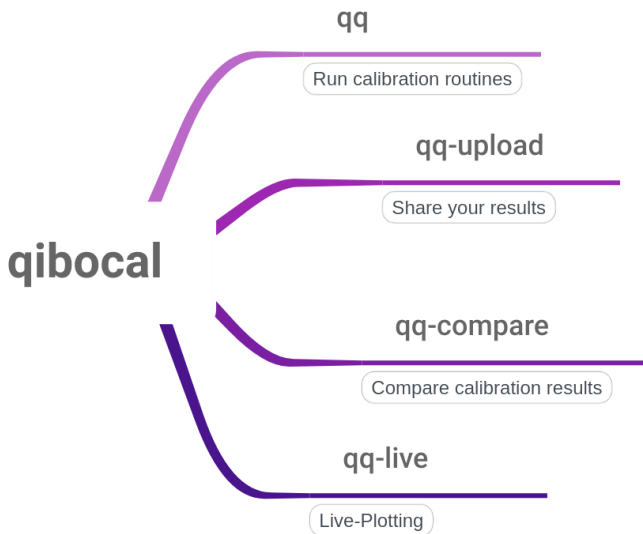
Uniform RB	Covered by Theorem 8	Discussed in Sec. VIC
<ul style="list-style-type: none">• Standard Clifford RB [5, 34]• Real RB [35]• Simultaneous RB [36]• dihedral RB* [37]• cNOT-dihedral RB [38]• Character RB* [39]• Restricted gate set RB [40]• Monomial RB [14]• Complete RB [41]• Leakage RB (1)** [19]• Leakage RB (2)** [42]• Unitarity RB** [16]• Loss RB** [18]• Measurement based RB [43]• Logical RB [44]• Pauli channel tomography* [45, 46]• Linear XEB* [29]	Interleaved RB <ul style="list-style-type: none">• Standard interleaved RB [4]• <i>T</i>-gate interleaved RB [47]• Iterative RB [48]• Individual gate RB [9]• Hybrid RB* [9]• Cycle RB* [13]	<ul style="list-style-type: none">• RB tomography [49]
	Nonuniform RB (Approximate) <ul style="list-style-type: none">• Approximate RB [14]• NIST RB [5, 50]	(Subset) <ul style="list-style-type: none">• Generator RB [14, 51]• Direct RB [15]• Coset (2-for-1) RB* [39] Covered by Theorem 10

Currently in Qibocal the following protocols are implemented:

- Standard RB
- Simultaneous Filtered RB
- Xld RB

We are currently developing a suite for the development of the **latest** quantum benchmarking protocols available in the **literature**.

REPORTING TOOLS



2303.10397

- Platform agnostic approach
- Launch calibration routines easily
- Live-plotting tools
- Autocalibration (under development)

Home

Timestamp
Summary

Actions

Ramsey - 0

Ramsey Experiment

Export to pdf

Platform: tii_rfso4x2
Run date: 2023-05-07
Start time (UTC): 05:20:51
End time (UTC): 05:21:19

Summary

In the table below we show the libraries and respective versions used in Ramsey Experiment.

Library	Version
numpy	1.23.5
qibo	0.1.13
qibocal	0.0.2
qibolab	0.0.3

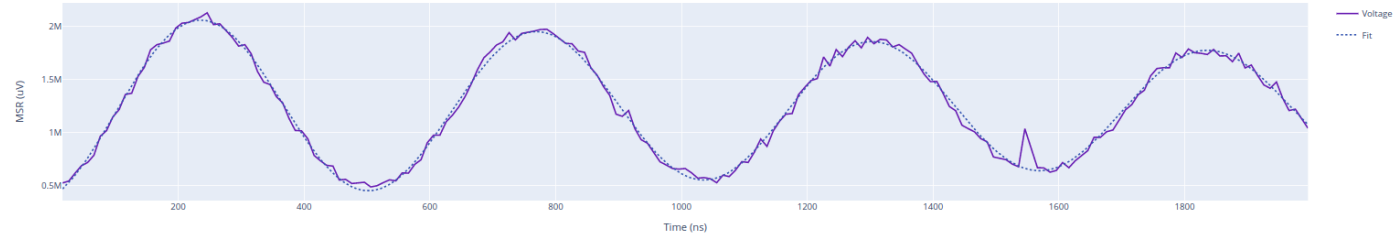
Actions

Please find below data generated by actions:

Ramsey - 0

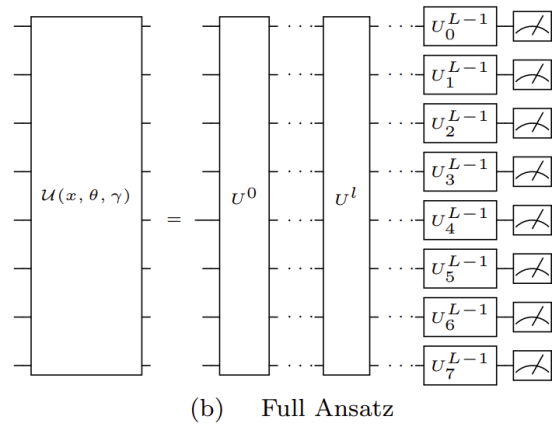
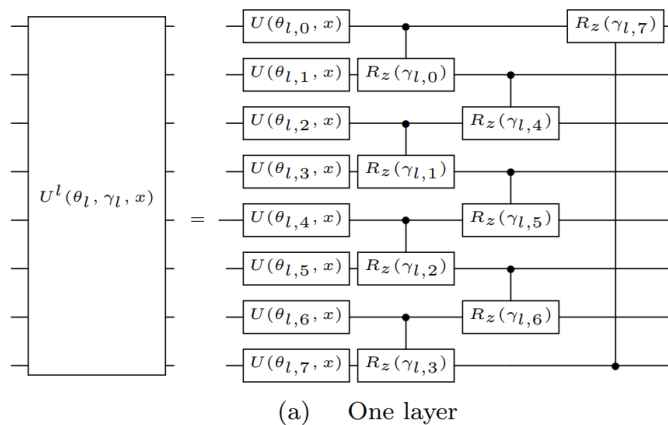
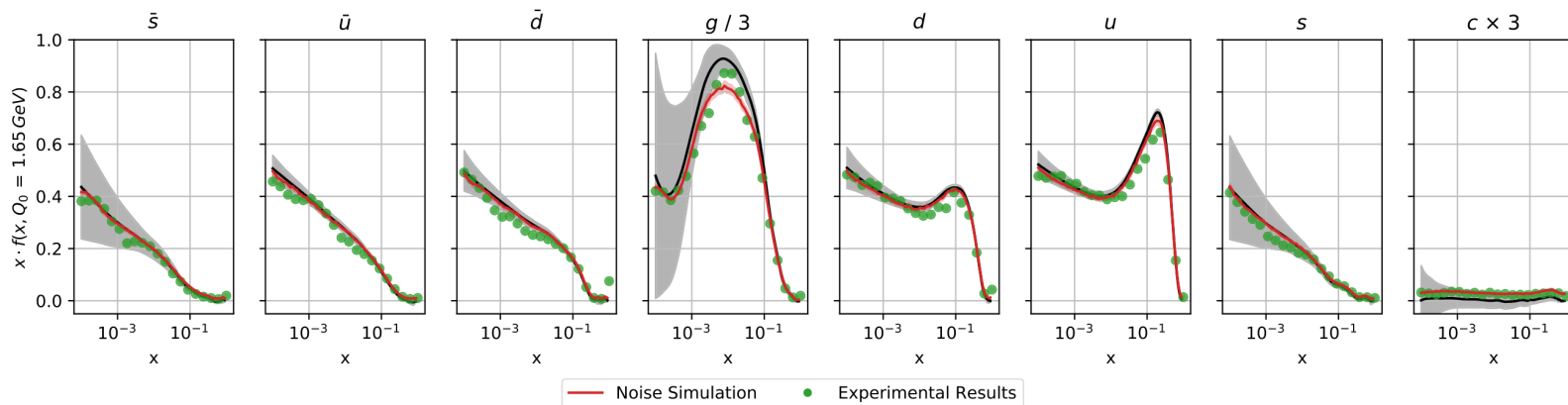
- Qubit 0

qubit	Fitting Parameter	Value
0	delta_frequency	-625,626.0 Hz
0	drive_frequency	5542303347.0 Hz

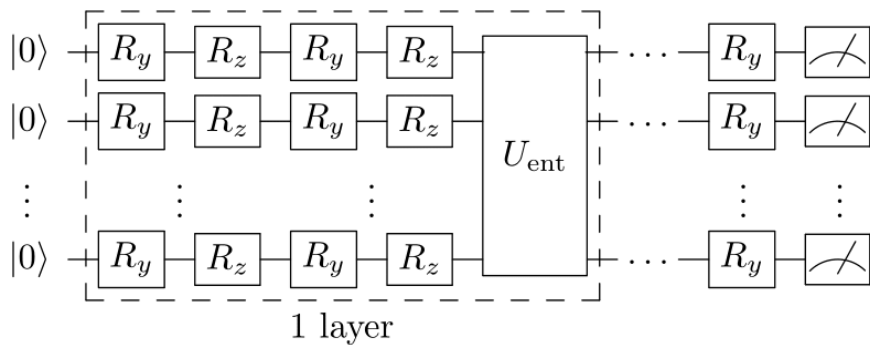


APPLICATIONS

DETERMINATION OF PARTON DISTRIBUTION FUNCTIONS USING QML 2011.13934

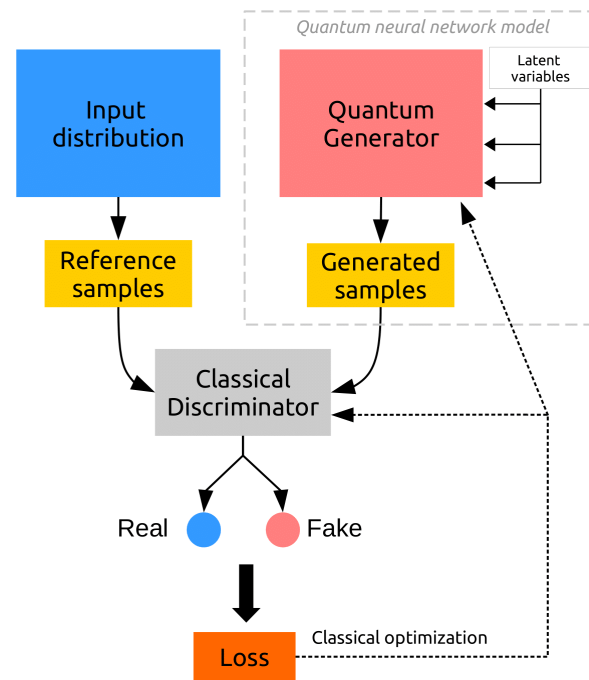


MONTECARLO EVENT GENERATOR USING QGAN



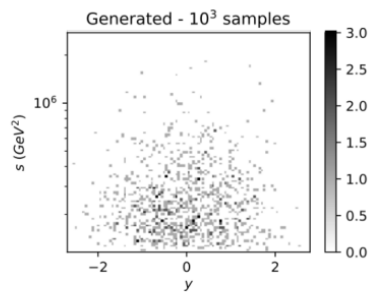
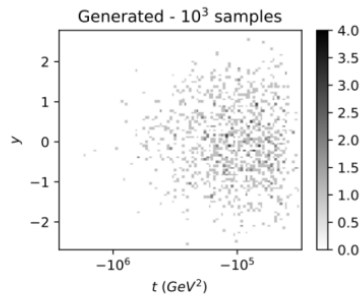
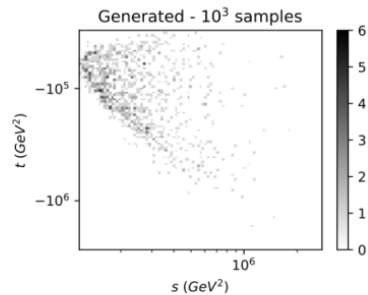
Style-based approach

$$R_{y,z}^{l,m}(\phi_g, z) = R_{y,z} \left(\phi_g^{(l)} z^{(m)} + \phi_g^{(l-1)} \right)$$

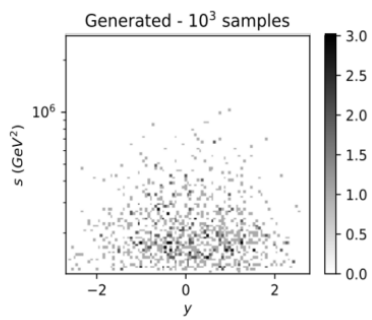
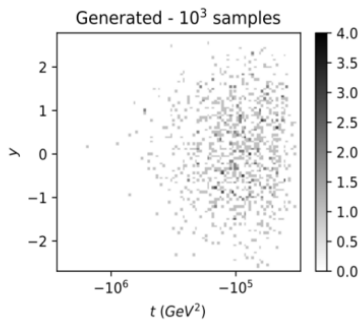
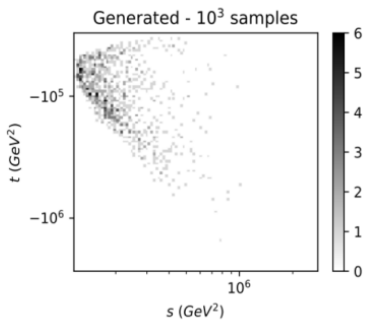


RESULTS WITH $pp \rightarrow t\bar{t}$

ionQ samples:



IBM Q samples:



Implementation largely hardware independent!

OUTLOOK

We have presented Qibo, a simple full stack API capable of doing

- High performance quantum simulation: qibojit
- Hardware control: qibolab
- Hardware calibration: qibocal

Why should you choose Qibo?

- Publicly available as an open source project
- Modular layout design with the possibility of adding
 - a new backend for simulation
 - a new platform for hardware control
- Community driven effort
- Many researchers are using it for HEP applications

The screenshot shows the GitHub repository for Qibo. At the top, it indicates the repository is public, has 24 forks, and 186 stars. Below this, there are buttons for 'Go to file', 'Add file', and 'Code'. A pull request by renatomello is highlighted, showing a merge of 87% from qiboteam/generalize_chan... with 6,244 commits. A table of recent commits follows, listing files like .github, doc, docker, examples, src/qibo, tests, .gitignore, pre-commit-config.yaml, .readthedocs.yml, LICENSE, README.md, poetry.lock, and pyproject.toml, along with their commit messages and timestamps. Below the commits, the README.md file is displayed, featuring the QIBO logo and a description of the project as an open-source full stack API for quantum simulation and hardware control. It lists key features such as a standard language for quantum circuits, a growing code-base of applications, efficient simulation backends, and a simple mechanism for implementing new backends. On the right side, there are sections for 'About' (describing Qibo as a framework for quantum computing with hardware acceleration), 'Releases' (showing the latest release Qibo 0.1.13), 'Packages' (listing the qibo package), 'Used by' (showing logos of organizations using Qibo), 'Contributors' (showing avatars of contributors), and 'Environments' (showing the github-pages environment).

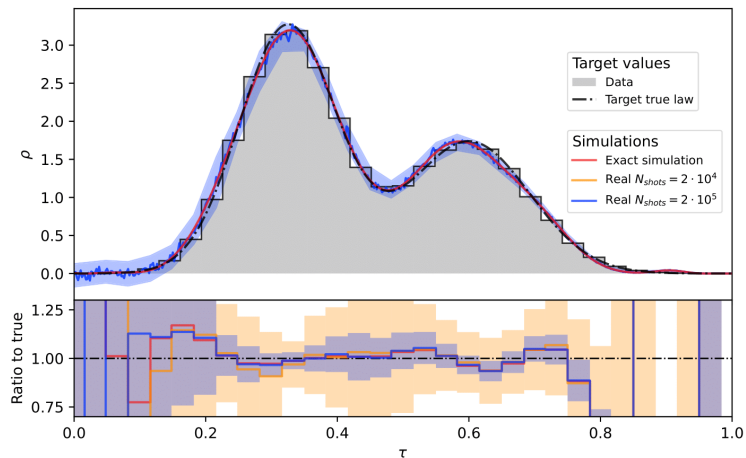
THANKS FOR LISTENING!

BACKUP SLIDES

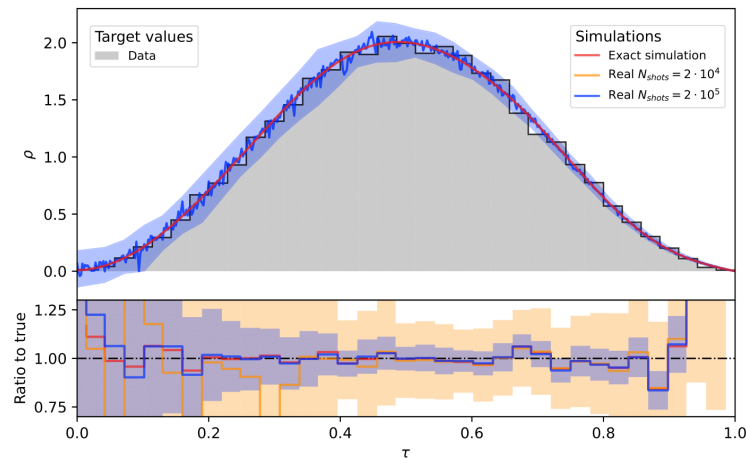
FITTING PDFs USING ADIABATIC EVOLUTION

Use quantum adiabatic machine learning for the determination of PDF and sampling.

PDF estimation - $\rho(x) = 0.6\mathcal{N}(x; -10, 4) + 0.4\mathcal{N}(x; 5, 5)$



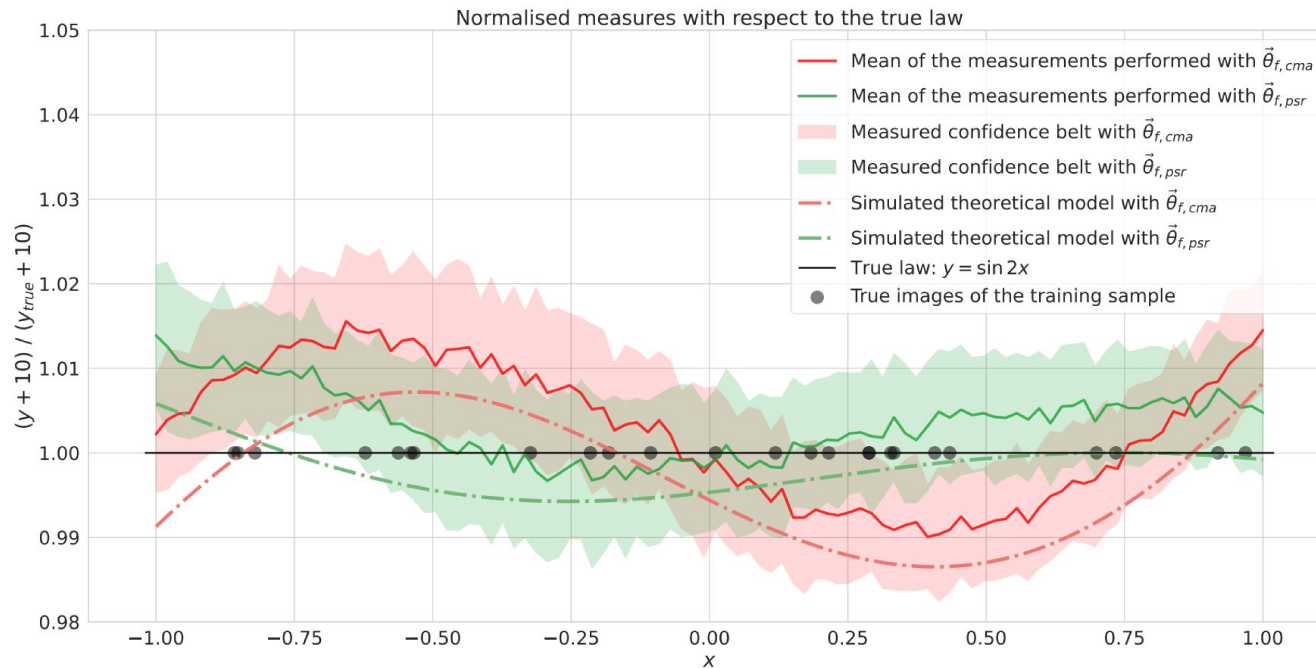
PDF estimation - y in $pp \rightarrow t\bar{t}$ decay



2303.11346

PARAMETER SHIFT RULE ON HARDWARE

Successfully performed a gradient descent on a QPU with a single using Parameter Shift Rule algorithm.



Scansionato con CamScanner