

# Full-stack Quantum Machine Learning for HEP

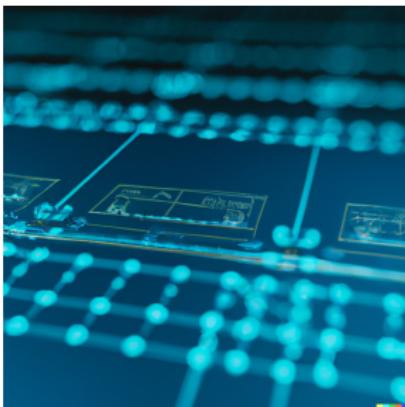
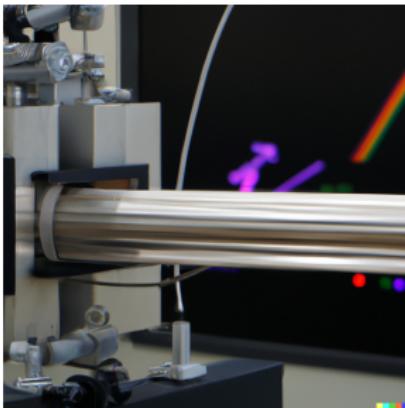
## NTU visit

---

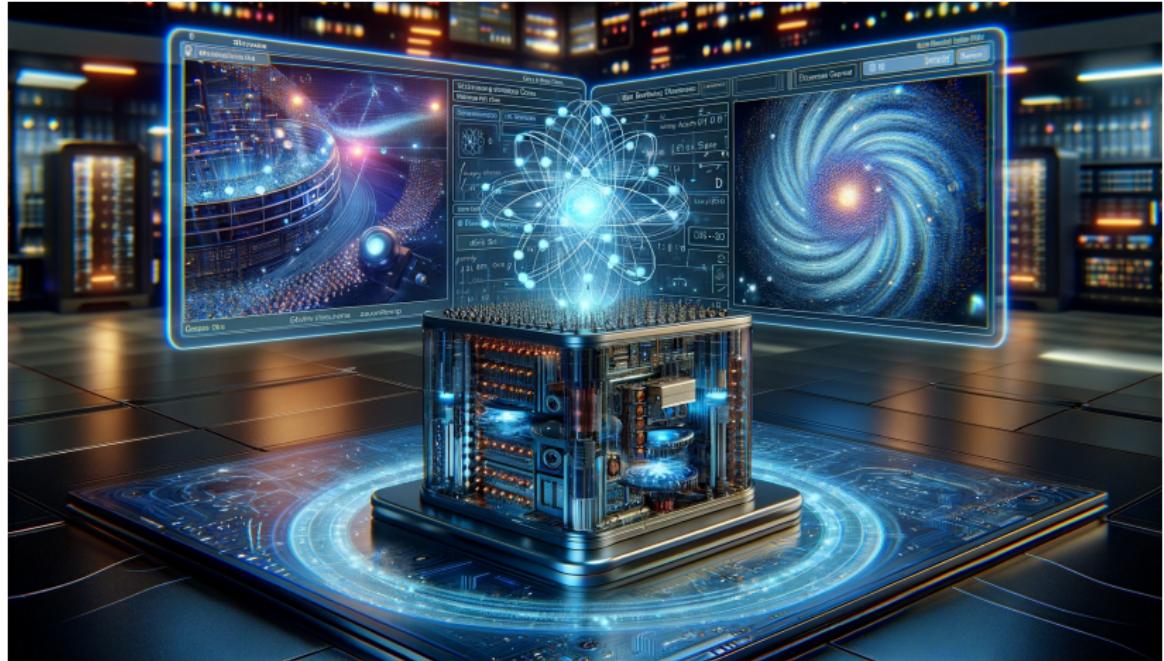
Matteo Robbiati  
26 January 2024



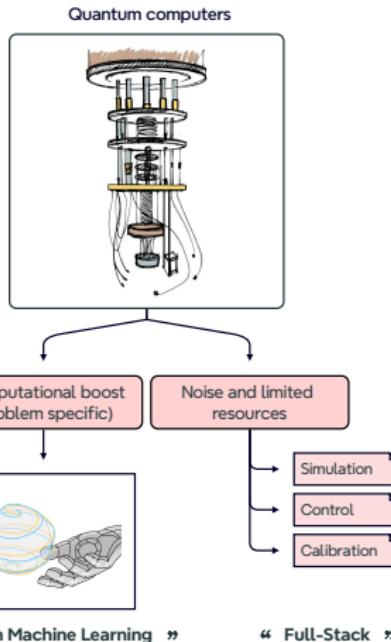
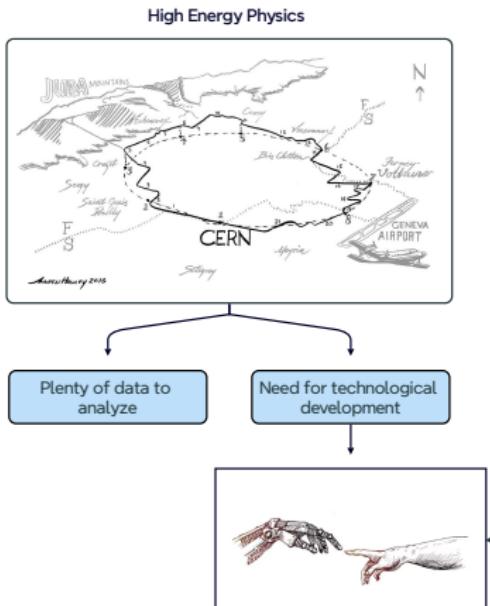
# DALL-E 2 explaining my title



# DALL-E 3 explaining my title



## Me explaining my title



## **Introductory concepts**

---

# Machine Learning (ML)

ML helps in solving statistical problems, such as data generation, classification, etc.

# Machine Learning (ML)

ML helps in solving statistical problems, such as data generation, classification, etc.

Considering the supervised ML approach:

ML helps in solving statistical problems, such as data generation, classification, etc.

Considering the supervised ML approach:

- ❖ we aim to know some hidden law between two variables:  $y = f(x)$ ;

ML helps in solving statistical problems, such as data generation, classification, etc.

Considering the supervised ML approach:

- ❖ we aim to know some hidden law between two variables:  $y = f(x)$ ;
- 📊 we define a parameteric model which returns  $\hat{y}_{\text{est}} = f_{\text{est}}(x; \theta)$ ;

ML helps in solving statistical problems, such as data generation, classification, etc.

Considering the supervised ML approach:

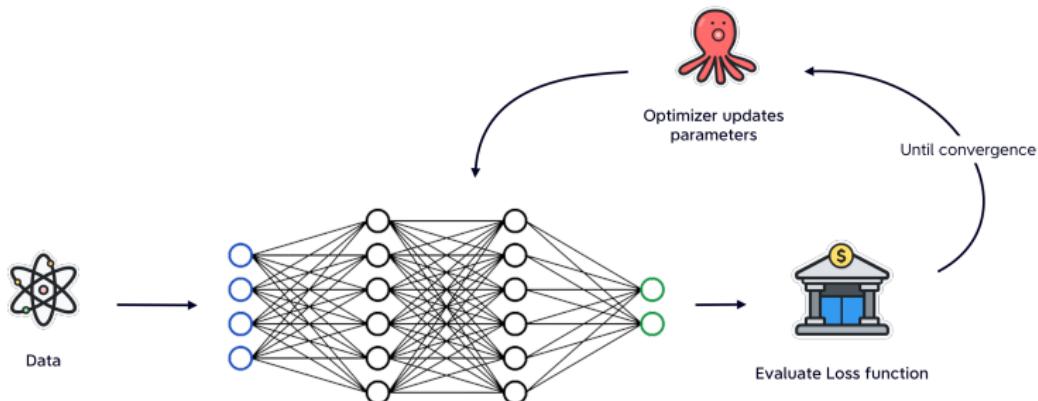
- ❖ we aim to know some hidden law between two variables:  $\mathbf{y} = f(\mathbf{x})$ ;
- 📊 we define a parameteric model which returns  $\mathbf{y}_{\text{est}} = f_{\text{est}}(\mathbf{x}; \boldsymbol{\theta})$ ;
- 🔭 we define an optimizer, which task is to compute  $\operatorname{argmin}_{\boldsymbol{\theta}} [J(\mathbf{y}_{\text{meas}}, \mathbf{y}_{\text{est}})]$ .

# Machine Learning (ML)

ML helps in solving statistical problems, such as data generation, classification, etc.

Considering the supervised ML approach:

- ❖ we aim to know some hidden law between two variables:  $y = f(x)$ ;
- 📊 we define a parametric model which returns  $\hat{y} = f_{\text{est}}(x; \theta)$ ;
- 🔭 we define an optimizer, which task is to compute  $\operatorname{argmin}_{\theta} [J(y_{\text{meas}}, \hat{y})]$ .

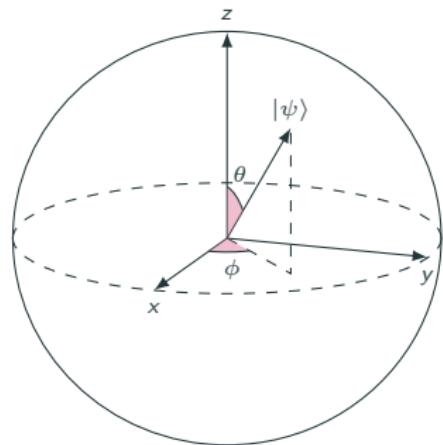


## Qubits (on the Bloch sphere)

## Qubits (on the Bloch sphere)

Qubits' states can be used to process information:

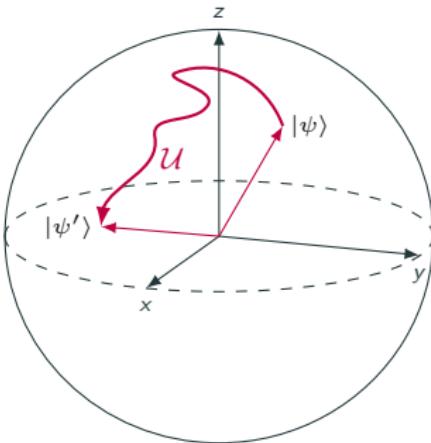
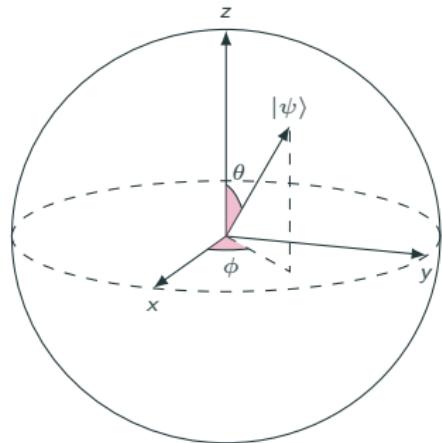
$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle \quad \text{where} \quad \alpha = \cos \frac{\theta}{2}, \quad \beta = e^{i\phi} \sin \frac{\theta}{2}.$$



## Qubits (on the Bloch sphere)

Qubits' states can be used to process information:

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle \quad \text{where} \quad \alpha = \cos \frac{\theta}{2}, \quad \beta = e^{i\phi} \sin \frac{\theta}{2}$$

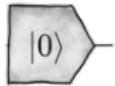


And this information can be manipulated applying unitaries  $\mathcal{U}$ .

# Parametrized Quantum Circuits

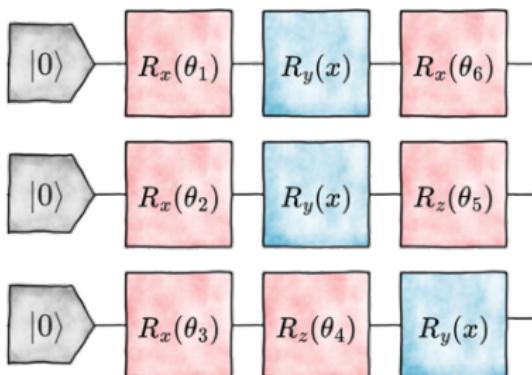
# Parametrized Quantum Circuits

- ☞ Classical bits are replaced by **qubits**:  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ ;



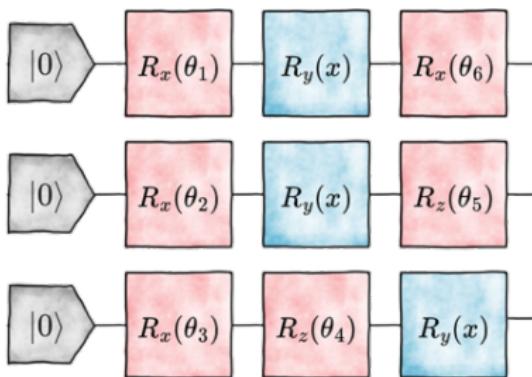
# Parametrized Quantum Circuits

- ✍ Classical bits are replaced by **qubits**:  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ ;
- ⚙️ we modify the qubits state by applying unitaries, which can be parametric  $\mathcal{U}(\theta)$ .



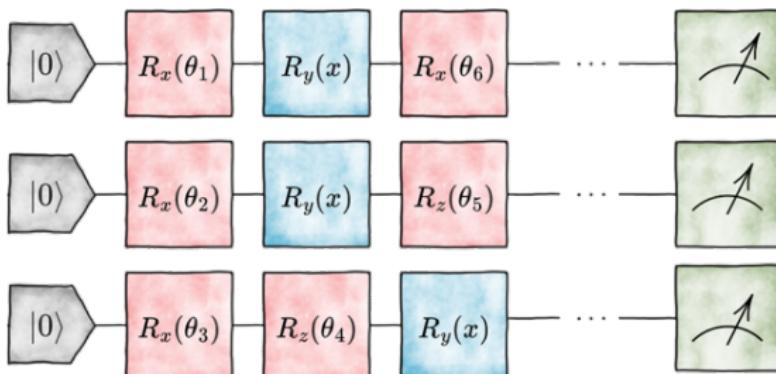
# Parametrized Quantum Circuits

- ✍ Classical bits are replaced by **qubits**:  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ ;
- ⚙️ we modify the qubits state by applying **unitaries**, which can be parametric  $\mathcal{U}(\theta)$ .
- ✍ we call the unitaries “**gates**” and many gates together “**circuit**”.



# Parametrized Quantum Circuits

- ✍ Classical bits are replaced by **qubits**:  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ ;
- ⚙️ we modify the qubits state by applying unitaries, which can be parametric  $\mathcal{U}(\theta)$ .
- ✍ we call the unitaries “**gates**” and many gates together “**circuit**”.
- 👁 after executing the circuit, information is accessed computing expectation values of target observables on the new qubits state.



## 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.

# Quantum Machine Learning - operating on qubits

## Machine Learning

$\mathcal{M}$ : model;

$\mathcal{O}$ : optimizer;

$\mathcal{J}$ : loss function.

$(x, y)$ : data

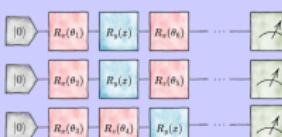
## Quantum Computation

$\mathcal{Q}$ : qubits;

$\mathcal{S}$ : superposition;

$\mathcal{E}$ : entanglement.

## Circuit execution



## Machine Learning

$\mathcal{M}$ : model;

$\mathcal{O}$ : optimizer;

$\mathcal{J}$ : loss function.

$(x, y)$ : data

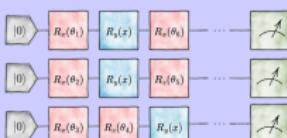
## Quantum Computation

$\mathcal{Q}$ : qubits;

$\mathcal{S}$ : superposition;

$\mathcal{E}$ : entanglement.

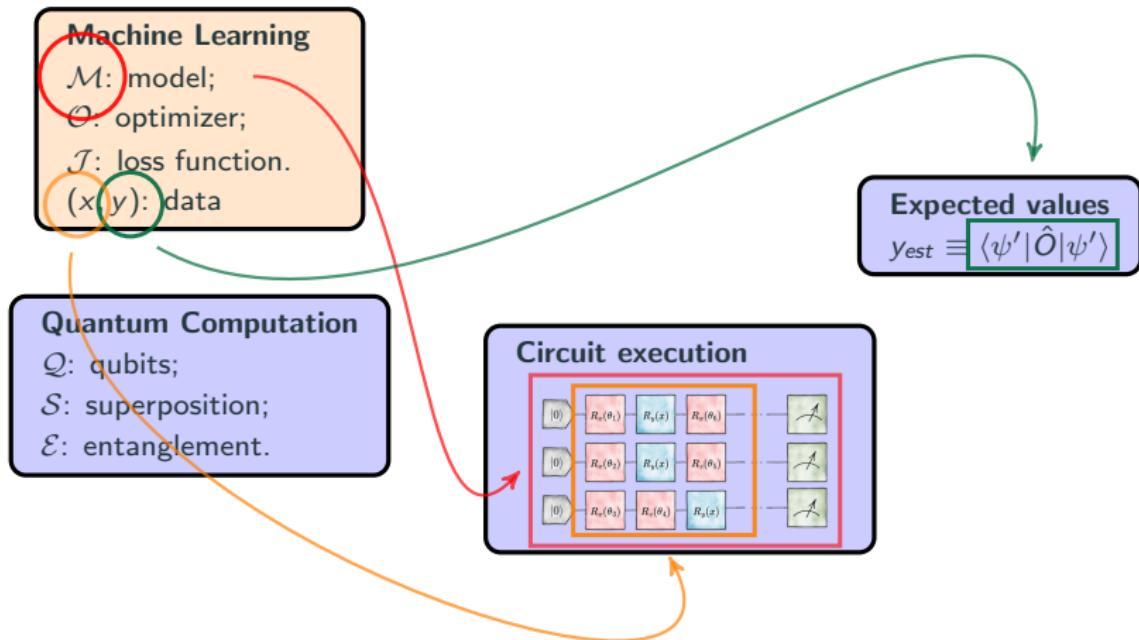
## Circuit execution



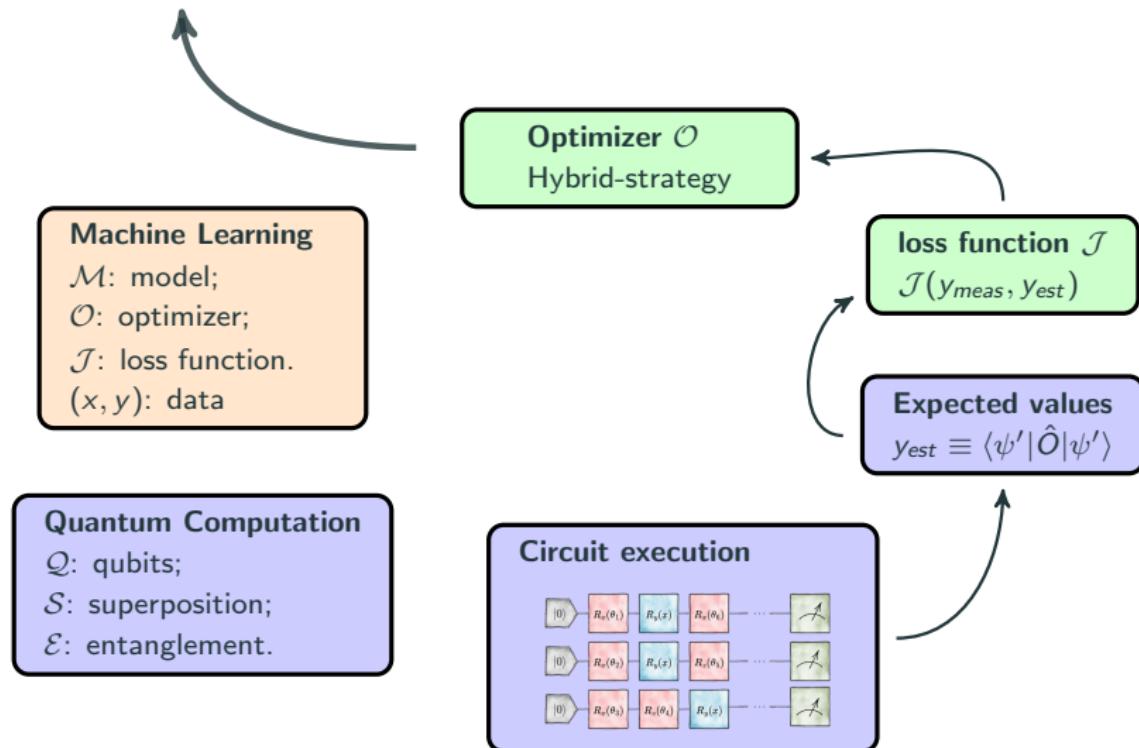
## Expected values

$$y_{est} \equiv \langle \psi' | \hat{O} | \psi' \rangle$$

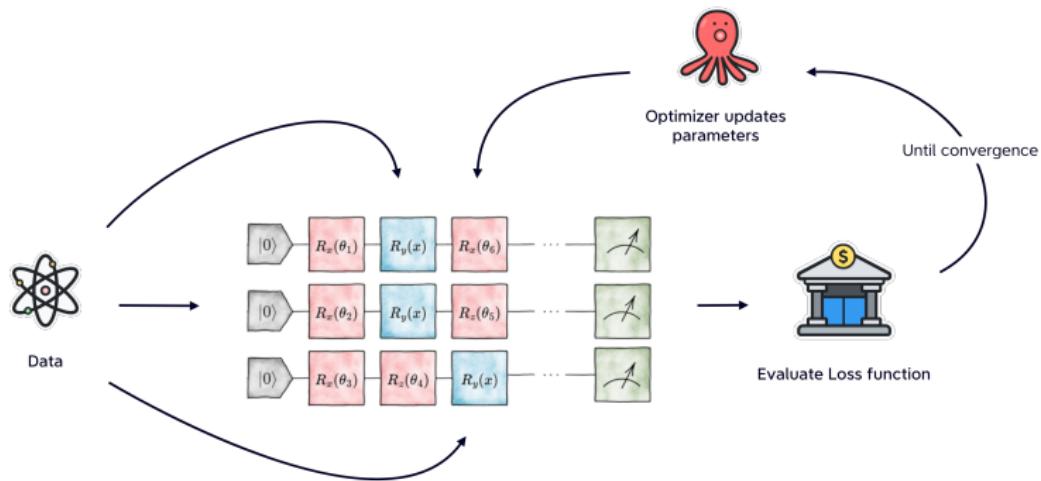
# Quantum Machine Learning - encoding the problem



# Quantum Machine Learning!



# From ML to QML



# Why QML?

## Why QML?

---

- ☛ we collect **high-dimensional** data,  
which challenge classical models;

## Why QML?

- ☛ we collect **high-dimensional** data, which challenge classical models;
- ❖ an  $N$ -long input variable can be stored in a  $\log N$  qubits system;

## Why QML?

- ⌚ we collect **high-dimensional** data, which challenge classical models;
- ❖ an  $N$ -long input variable can be stored in a  $\log N$  qubits system;
- ⚡ low power consumption<sup>1</sup>;

---

<sup>1</sup>  *Are quantum computers really energy efficient?*, Sophia Chen, Jun 2023

## Why QML?

- ☒ we collect **high-dimensional** data,  
which challenge classical models;
- ☒ an  $N$ -long input variable can be  
stored in a  $\log N$  qubits system;
- ☒ low power consuption<sup>1</sup>;
- ☒ can **superposition** and **entanglement**  
be exploited to use less **parameters**?

---

<sup>1</sup>☒ Are quantum computers really energy efficient?, Sophia Chen, Jun 2023

# Why QML?

- ⌚ we collect **high-dimensional** data, which challenge classical models;
- ❖ an  $N$ -long input variable can be stored in a log  $N$  qubits system;
- ⚡ low power consumption<sup>1</sup>;
- ☒ can **superposition** and **entanglement** be exploited to use less **parameters**?
- ⌚ can superposition and entanglement better deal with quantum data?<sup>23</sup>

---

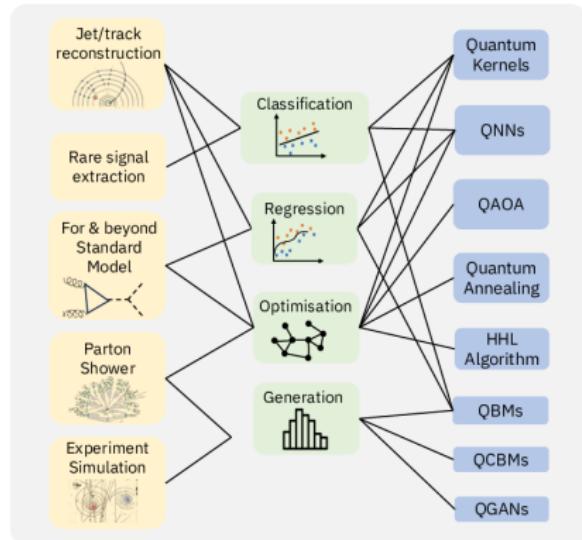
<sup>1</sup> Почемущи: Are quantum computers really energy efficient?, Sophia Chen, Jun 2023

<sup>2</sup> Почемущи: Quantum Computing for High-Energy Physics: State of the Art and Challenges, A. Di Meglio et al., Jul 2023

<sup>3</sup> Почемущи: Quantum anomaly detection in the latent space of proton collision events at the LHC, K. A. Woźniak et al., Jan 2023

# Why QML?

- 🛒 we collect **high-dimensional** data, which challenge classical models;
- 🕸️ an  $N$ -long input variable can be stored in a log  $N$  qubits system;
- 🌿 low power consumption<sup>1</sup>;
- ☒ can **superposition** and **entanglement** be exploited to use less **parameters**?
- 🎒 can superposition and entanglement better deal with quantum data?<sup>23</sup>

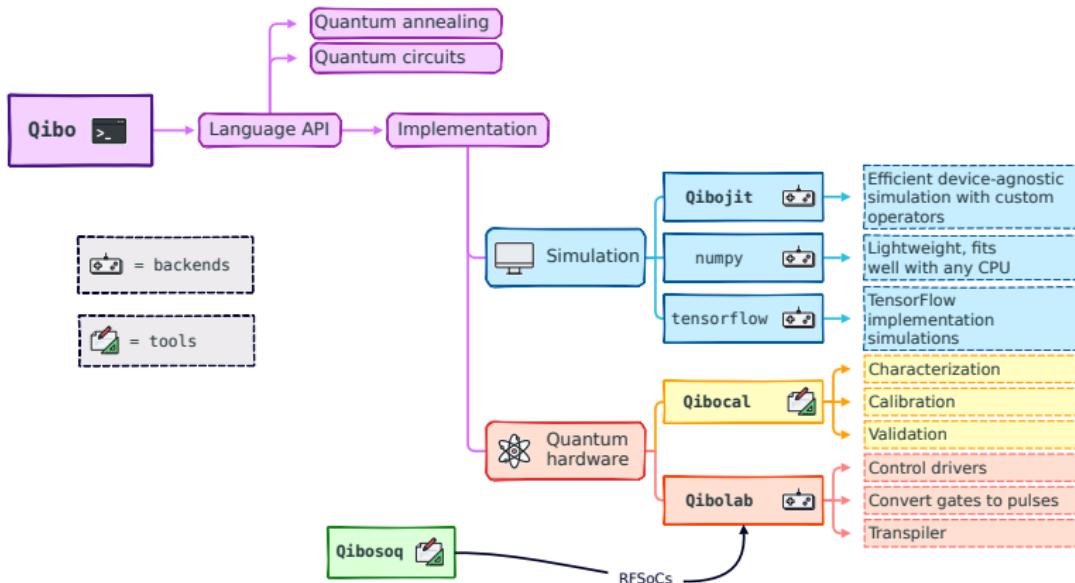


<sup>1</sup> 📈 Are quantum computers really energy efficient?, Sophia Chen, Jun 2023

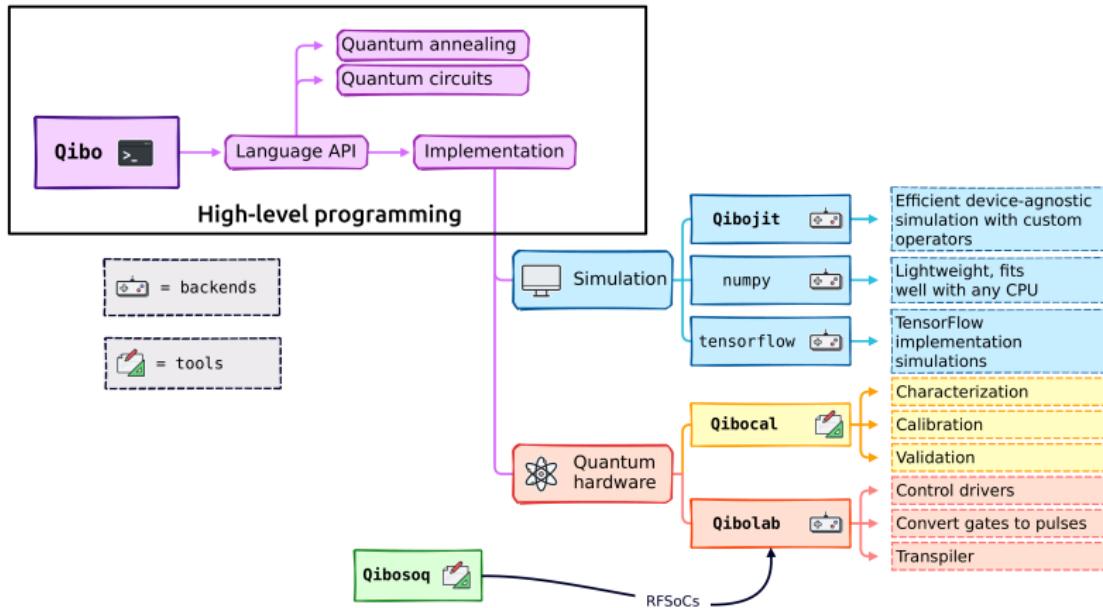
<sup>2</sup> 📈 Quantum Computing for High-Energy Physics: State of the Art and Challenges, A. Di Meglio et al., Jul 2023

<sup>3</sup> 📈 Quantum anomaly detection in the latent space of proton collision events at the LHC, K. A. Woźniak et al., Jan 2023

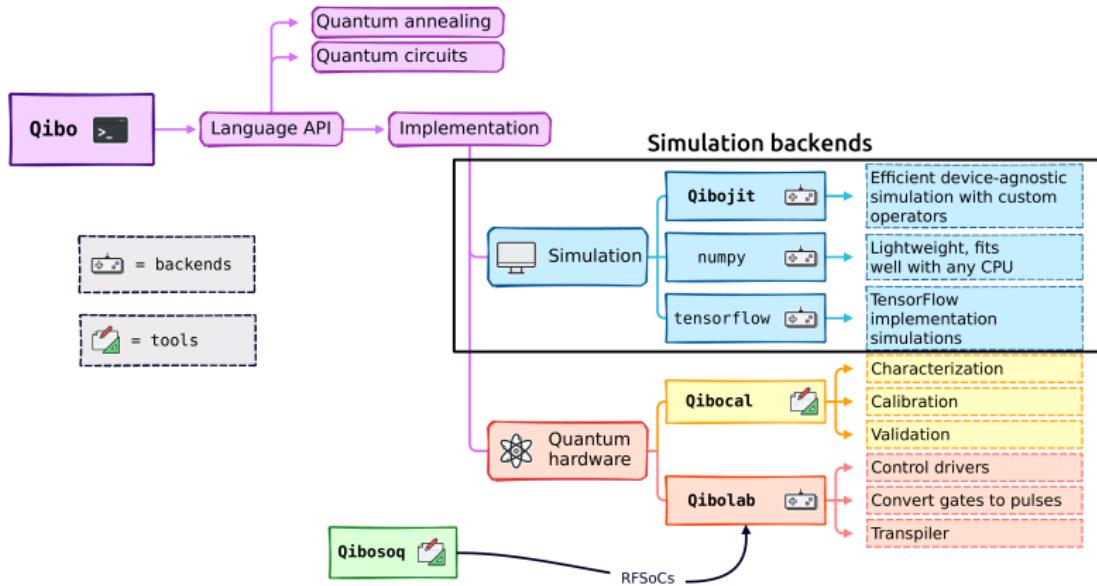
# My playground: Qibo



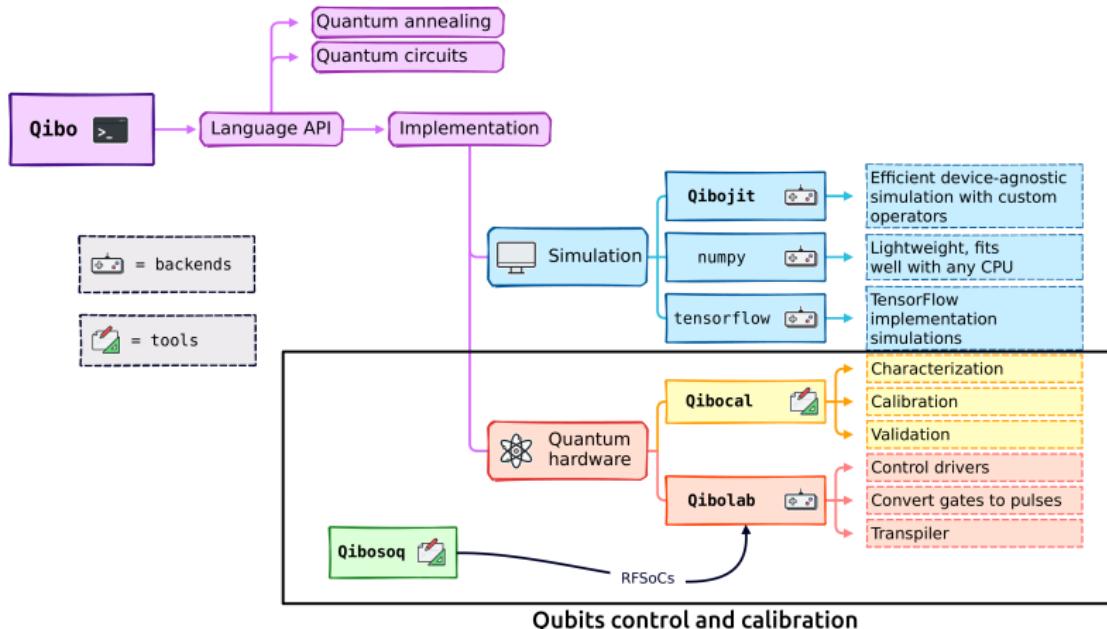
# My playground: Qibo



# My playground: Qibo



# My playground: Qibo

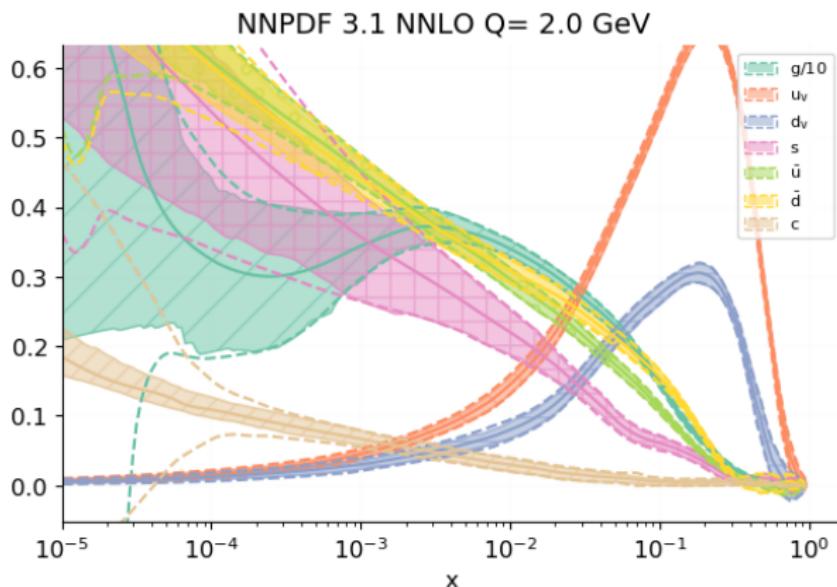


**A target from HEP**

---

## Parton Distribution Functions as QML target

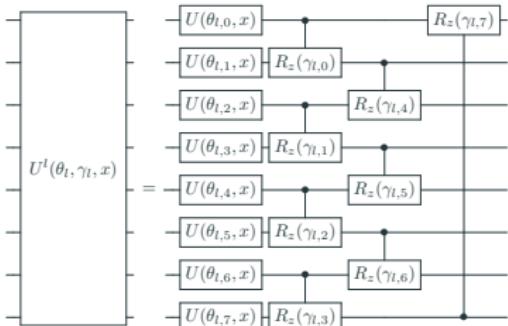
Taking one parton  $p_i$ , we can see its PDF as the probability for  $p_i$  of carrying a fraction  $x$  of the momentum of the proton.





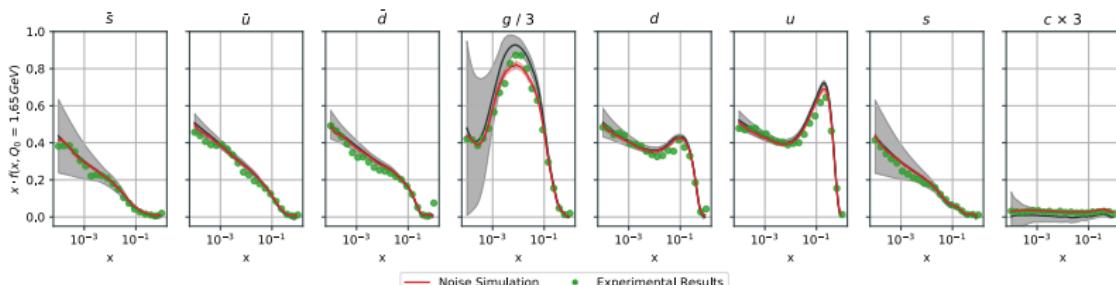
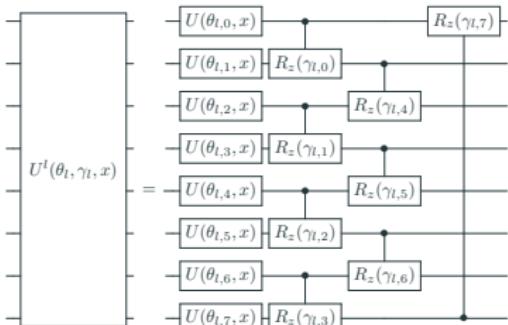
- Define a circuit  $\mathcal{U}(x; \theta)$  using one qubit per parton  $p_i$ ;
- fill gates with both  $x_i$  and  $\log(x_i)$ ;
- Compute PDF<sub>*i*</sub> prediction using expectation of  $Z_i = \bigotimes_{j=0}^n Z^{\delta_{ij}}$ :

$$\text{qPDF}_i(x_i; Q_0, \theta) = \frac{1 - \langle 0 | \mathcal{U}^\dagger Z_i \mathcal{U} | 0 \rangle}{1 + \langle 0 | \mathcal{U}^\dagger Z_i \mathcal{U} | 0 \rangle}$$



- Define a circuit  $\mathcal{U}(x; \theta)$  using one qubit per parton  $p_i$ ;
- fill gates with both  $x_i$  and  $\log(x_i)$ ;
- Compute PDF<sub>*i*</sub> prediction using expectation of  $Z_i = \bigotimes_{j=0}^n Z^{\delta_{ij}}$  :

$$\text{qPDF}_i(x_i; Q_0, \theta) = \frac{1 - \langle 0 | \mathcal{U}^\dagger Z_i \mathcal{U} | 0 \rangle}{1 + \langle 0 | \mathcal{U}^\dagger Z_i \mathcal{U} | 0 \rangle}$$



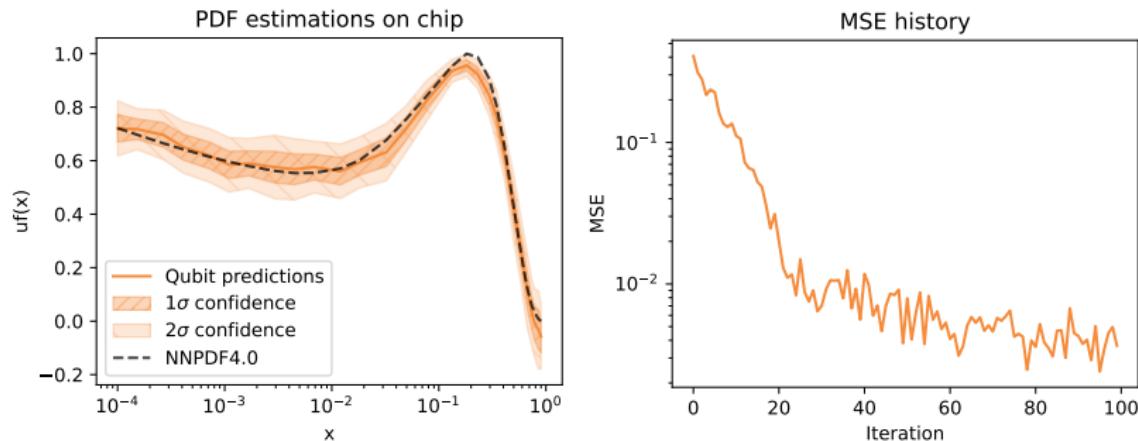


- ❖ We take into account the  $u$ -quark PDF;

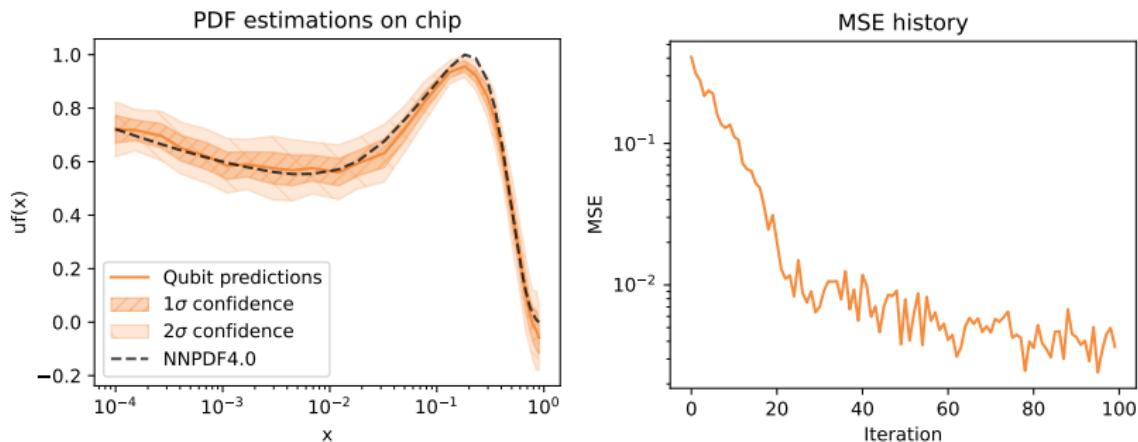
- ❖ We take into account the  $u$ -quark PDF;
- ❖ we pre-process it to fit the range  $[0, 1]$ ;

- ❖ We take into account the  $u$ -quark PDF;
- ❖ we pre-process it to fit the range  $[0, 1]$ ;
- ❖ we use Qibo, Qibolab and Qibocal to run a gradient descent.

- We take into account the  $u$ -quark PDF;
- we pre-process it to fit the range  $[0, 1]$ ;
- we use Qibo, Qibolab and Qibocal to run a gradient descent.

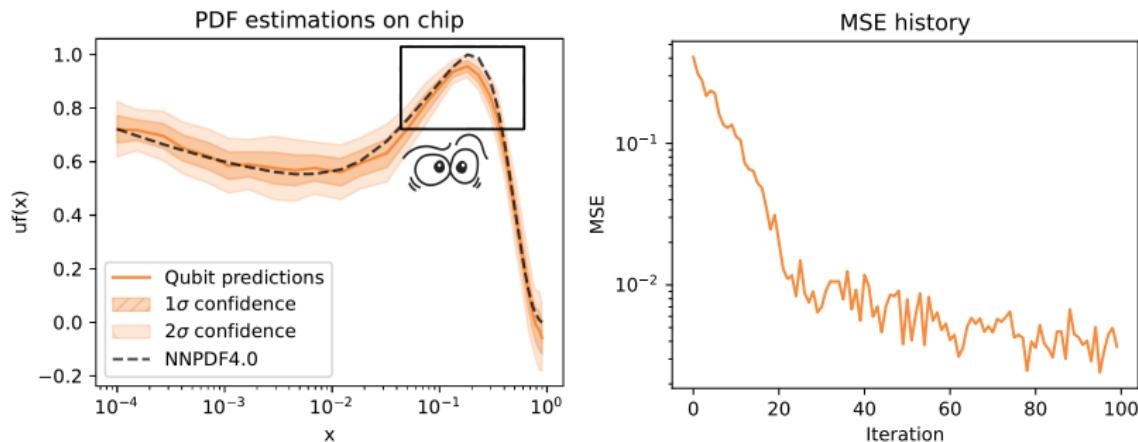


- We take into account the  $u$ -quark PDF;
- we pre-process it to fit the range  $[0, 1]$ ;
- we use Qibo, Qibolab and Qibocal to run a gradient descent.



Parameter	$N_{\text{train}}$	$N_{\text{params}}$	Optimizer	$N_{\text{shots}}$	$MSE_{\text{final}}$	$T_{\text{exe}}$
Value	30	14	Adam	250	$3.6 \cdot 10^{-3}$	78'

- We take into account the  $u$ -quark PDF;
- we pre-process it to fit the range  $[0, 1]$ ;
- we use Qibo, Qibolab and Qibocal to run a gradient descent.



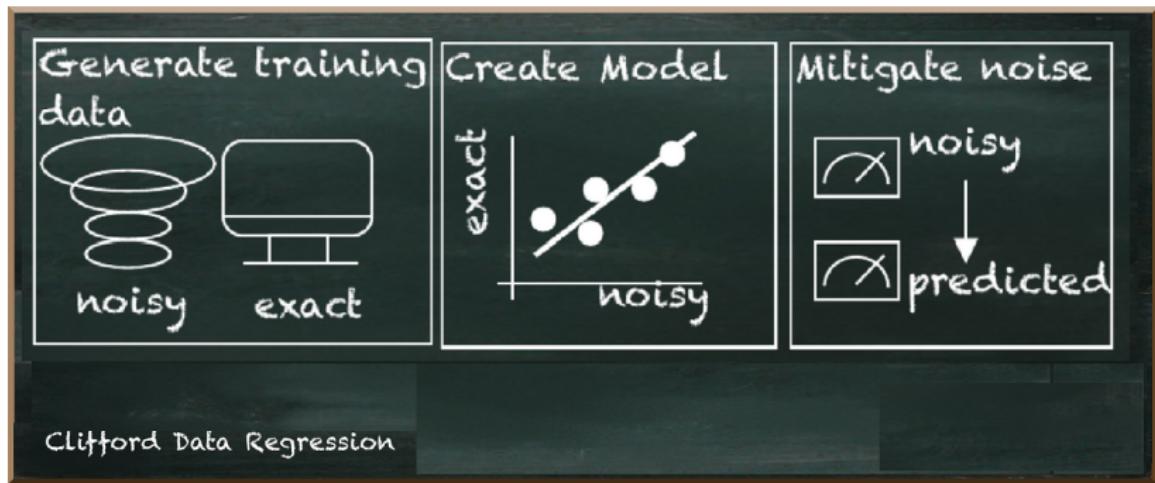
Parameter	$N_{\text{train}}$	$N_{\text{params}}$	Optimizer	$N_{\text{shots}}$	$MSE_{\text{final}}$	$T_{\text{exe}}$
Value	30	14	Adam	250	$3.6 \cdot 10^{-3}$	78'

💡 Idea: learning a noise map  $\ell$  and use it to clean expectation values from noise:

$$\langle \mathcal{O} \rangle_{\text{clean}} = \ell[\langle \mathcal{O} \rangle_{\text{noisy}}]$$

💡 Idea: learning a noise map  $\ell$  and use it to clean expectation values from noise:

$$\langle \mathcal{O} \rangle_{\text{clean}} = \ell[\langle \mathcal{O} \rangle_{\text{noisy}}]$$

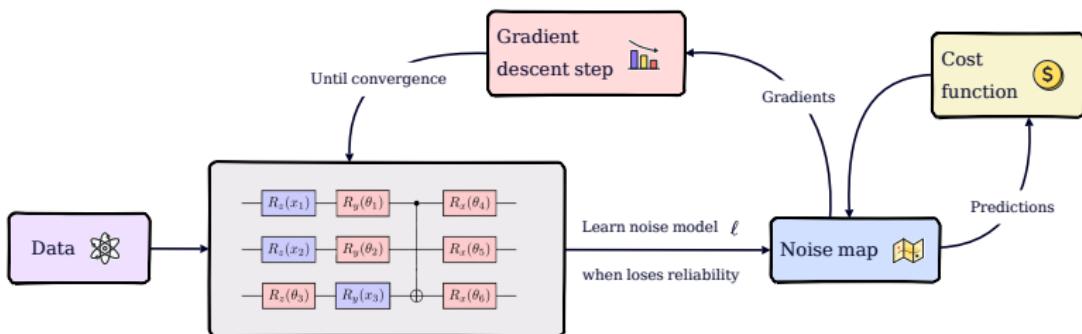


Credits: Frank Zickert.

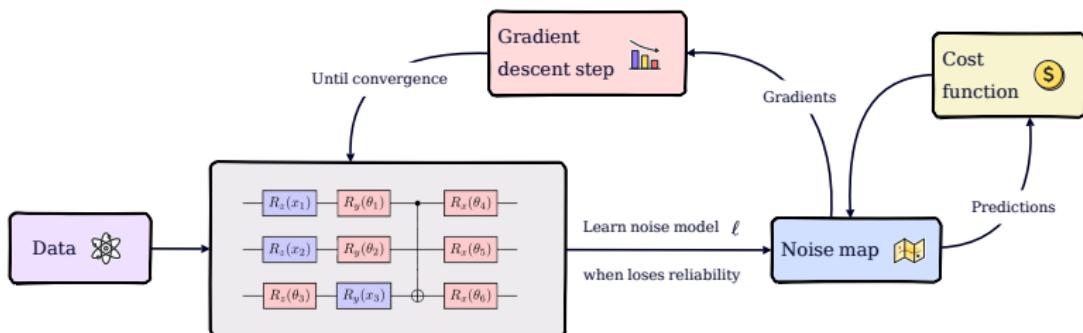


We try to mitigate both gradients and predictions at each optimization iteration.

We try to mitigate both gradients and predictions at each optimization iteration.

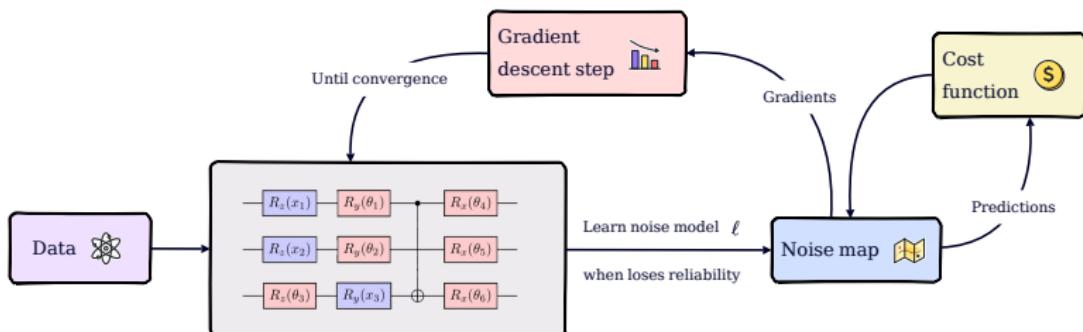


We try to mitigate both gradients and predictions at each optimization iteration.



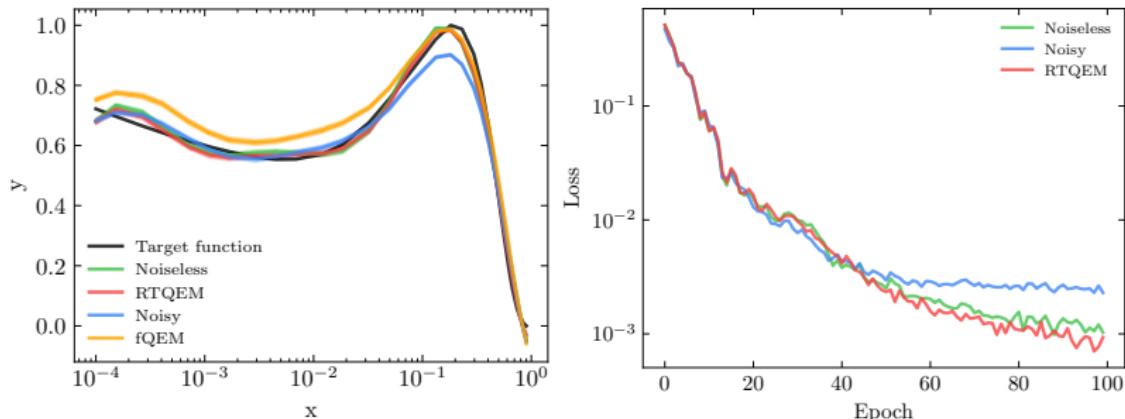
- ⌚ these experiments take long execution times (but we have Qibo!);

We try to mitigate both gradients and predictions at each optimization iteration.



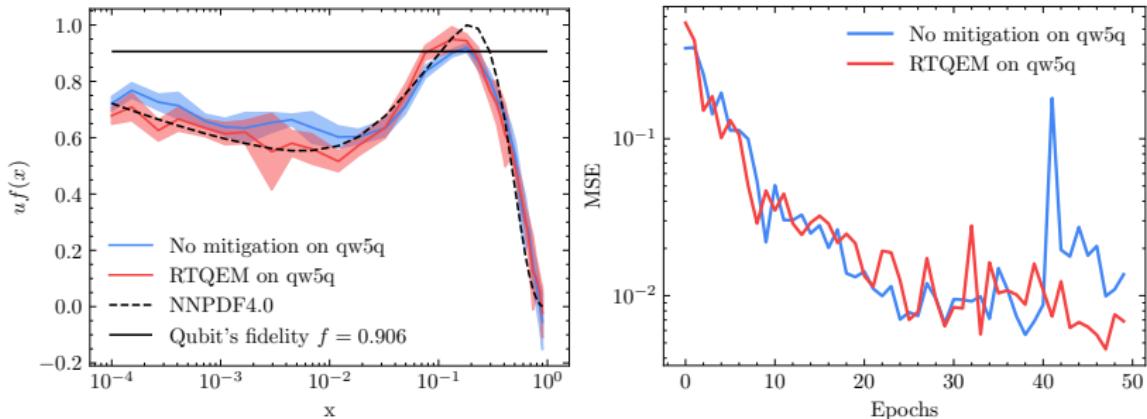
- ⌚ these experiments take long execution times (but we have Qibo!);
- ⚡ in some regimes, we aim to remove the bounds imposed by noise.

Parameter	$N_{\text{train}}$	$N_{\text{params}}$	$N_{\text{shots}}$	$\text{MSE}_{\text{rtqem}}$	$\text{MSE}_{\text{nomit}}$	Noise
Value	30	16	$10^4$	0.008	0.018	local Pauli



1. thanks to the RTQEM procedure, we reach a good minimum of the cost function;
2. the QEM is not effective if applied to a corrupted scenario (orange curve).

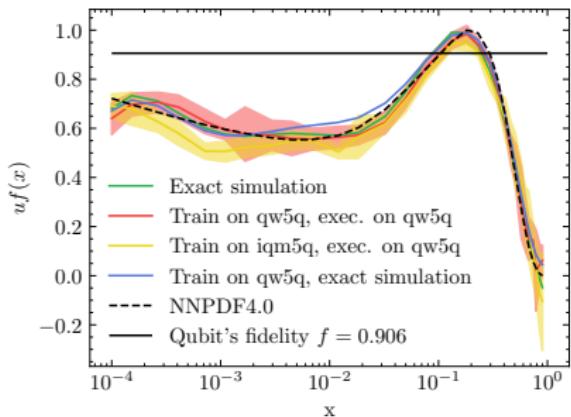
Parameter	$N_{\text{train}}$	$N_{\text{params}}$	$N_{\text{shots}}$	$\text{MSE}_{\text{rtqem}}$	$\text{MSE}_{\text{nomit}}$	Noise
Value	15	16	500	0.0042	0.0055	real noise



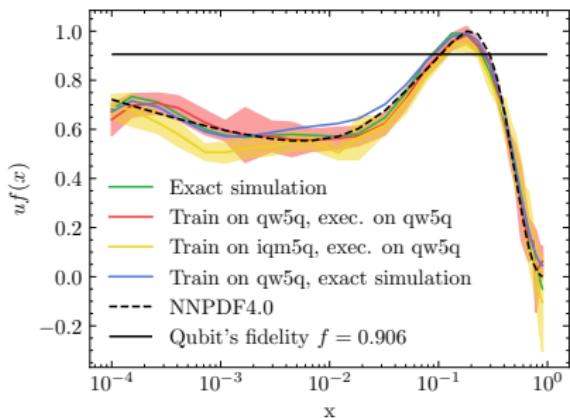
RTQEM allows exceeding the natural bound imposed by noise.

We perform a longer training on two different devices (and noises!) using the same initial conditions of the previous slide but  $N_{\text{epochs}} = 100$ .

We perform a longer training on two different devices (and noises!) using the same initial conditions of the previous slide but  $N_{\text{epochs}} = 100$ .

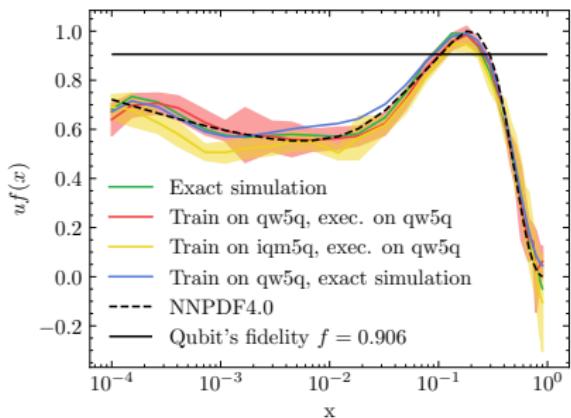


We perform a longer training on two different devices (and noises!) using the same initial conditions of the previous slide but  $N_{\text{epochs}} = 100$ .



- ⚙️ `qw5q` from QuantWare and controlled using Qblox instruments;
- ⚙️ `iqm5q` from IQM and controlled using Zurich Instruments.

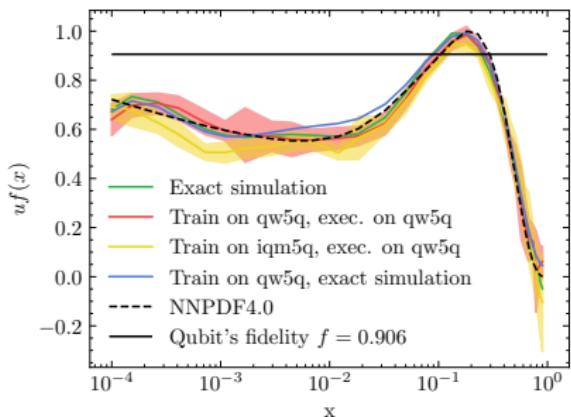
We perform a longer training on two different devices (and noises!) using the same initial conditions of the previous slide but  $N_{\text{epochs}} = 100$ .



- ⚙️ qw5q from QuantWare and controlled using Qblox instruments;
- ⚙️ iqm5q from IQM and controlled using Zurich Instruments.

Train.	Epochs	Pred.	Config.	MSE
qw5q	50	qw5q	noisy	0.0055
qw5q	50	qw5q	RTQEM	0.0042
qw5q	100	qw5q	RTQEM	0.0013
iqm5q	100	qw5q	RTQEM	0.0037
qw5q	100	sim	RTQEM	0.0016

We perform a longer training on two different devices (and noises!) using the same initial conditions of the previous slide but  $N_{\text{epochs}} = 100$ .



- ⚙️ qw5q from QuantWare and controlled using Qblox instruments;
- ⚙️ iqmq5q from IQM and controlled using Zurich Instruments.

Train.	Epochs	Pred.	Config.	MSE
qw5q	50	qw5q	noisy	0.0055
qw5q	50	qw5q	RTQEM	0.0042
qw5q	100	qw5q	RTQEM	0.0013
iqmq5q	100	qw5q	RTQEM	0.0037
qw5q	100	sim	RTQEM	0.0016

All the hardware results are obtained deploying the  $\theta_{\text{best}}$  on qw5q.

