DESY. QUANTUM Center for Quantum Technology and Applications

MATTER AND THE UNIVERSE

Precise Quantum Angle Generator Designed for Noisy Quantum Devices

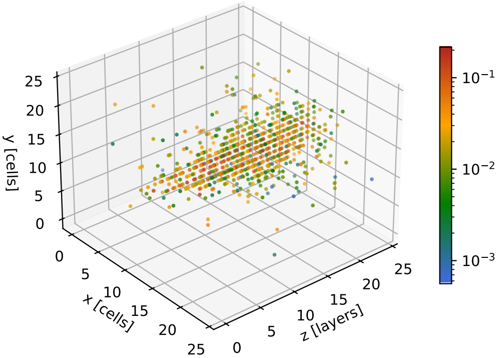
S. Monaco[1,2], F. Rehm[3], K. Borras[1,2], D. Krücker[1], S. Schnake[1,2]

[1]DESY, [2]RWTH Aachen University, [3]CERN

December 12, 2024

Quantum Computing for High Energy Physics

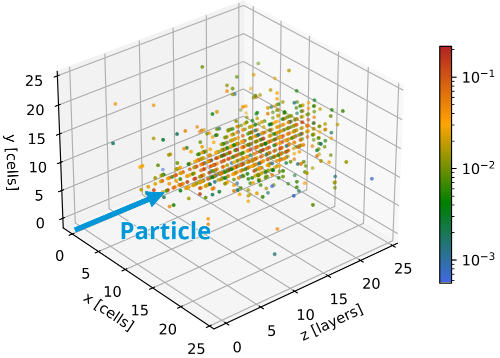
A brief motivation for Quantum Models in High Energy Physics.



Quantum Computing for High Energy Physics

Introduction

- **Use Case:** Particle Physics Calorimeter Simulations
  - Calorimeter detectors responsible for measuring particle energies in physics
  - Current Geant4 Monte Carlo simulations are **computationally demanding**
    - ➔ Searching for alternatives

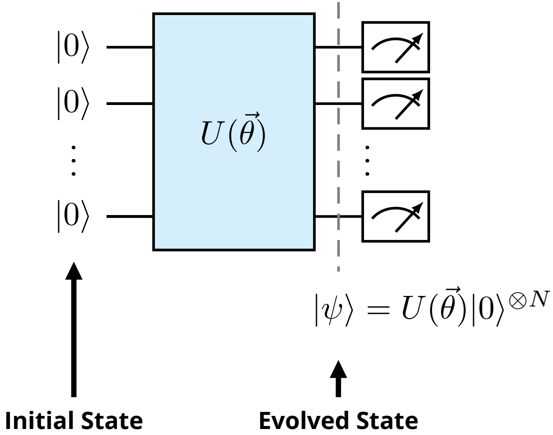


Previously: **Geant4 Monte Carlo Simulations**

Now: **Deep Learning** (150 000x speed up)

➔ Developed a Deep Learning model for calorimeter simulation which requires fewer computing resources (DLGAN)

Next: Explore **Quantum Computing**



Quantum Computing **allows for the accurate evolution of a quantum state** $|0\rangle^{\otimes N}$ into another $|\psi\rangle$

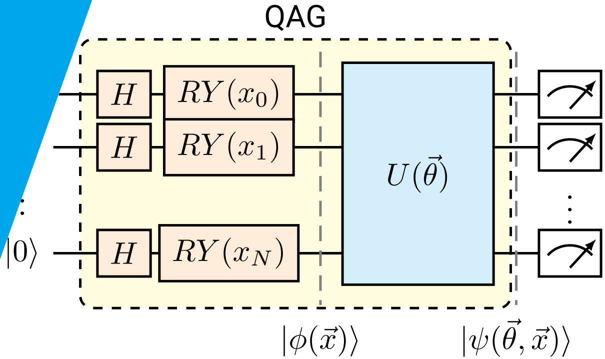
✓ High dimensional search space *Hilbert space*

- Fewer parameters needed
- Faster learning

compared to classical models

QAG Model

How the Quantum Angle Generator works and is trained

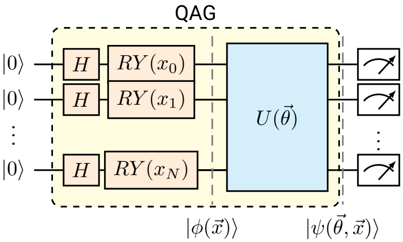


QAG Model
Introduction

In this study: A new quantum generative model:
Quantum Angle Generator (QAG)

Why a new model?

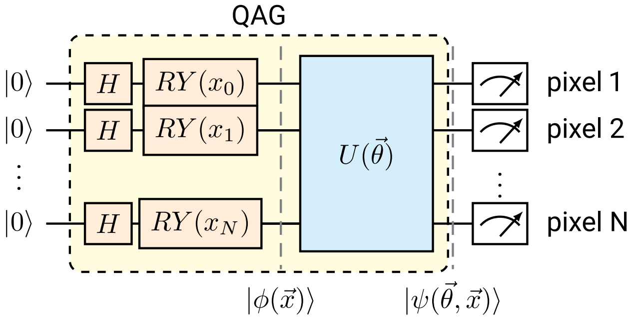
- Current quantum models do not satisfy our requirements
  - **QGAN** (Quantum Generative Adversarial Network)**:**
    Training is resource inefficient and unstable
  - **QCBM** (Quantum Circuit Born Machine)**:**
    Does not scale well in qubits and gates



QAG Model

Model Schema

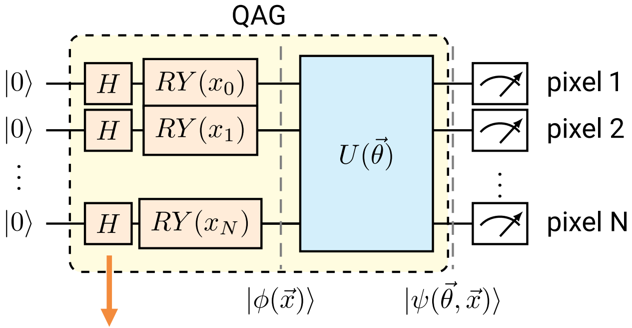
- The generation of N pixels requires N qubits



QAG Model

Model Schema

- The generation of N pixels requires N qubits



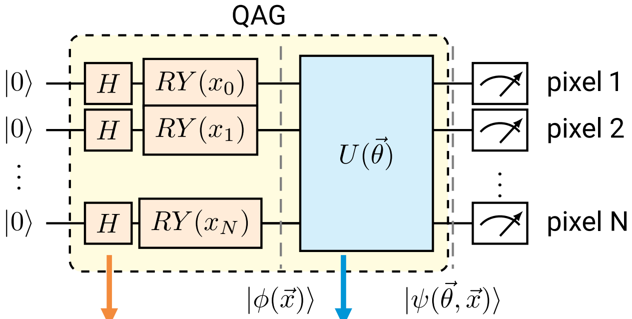
Quantum State Preparation

- Implement superposition through H
- Implement random noise through RY

QAG Model

Model Schema

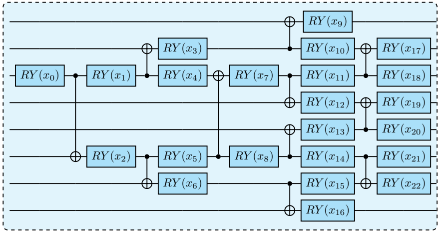
- The generation of N pixels requires N qubits



Quantum State Preparation

- Implement superposition through H
- Implement random noise through RY

Parametrized Unitary $U(\vec{\theta})$

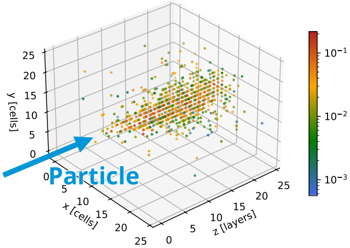


QAG Model

Downsampling

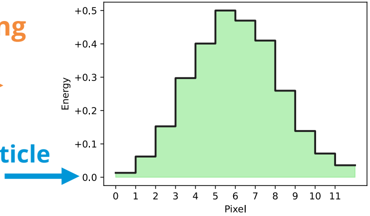
- Only initial investigation with simplified models
- Understand advantages and challenges

$(25 \times 25 \times 51)$



Downsampling

$(1 \times 1 \times 12)$



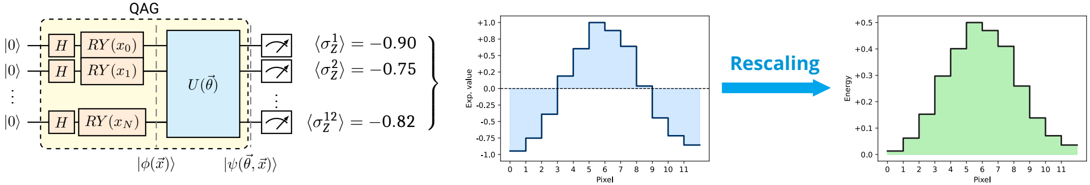
QAG Model

Generation process

Once the optimal parameters $\vec{\theta}^{*}$ are found through the training process

1. Generate a random noise-vector $\vec{x}$ for the RY gates
2. Compute expectation value $\langle \sigma_Z \rangle$ for each qubit:

$$\langle \sigma_Z \rangle = 2 * \frac{\#|0\rangle}{n_{\text{shots}}}$$



QAG Model

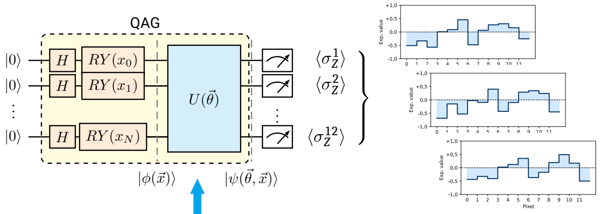
Training process

Choose a random initial set of parameters $\vec{\theta}_0$

For each epoch i

1. Generate M images
2. Evaluate the *MMD* loss
3. Update the parameters $\theta_i \rightarrow \theta_{i+1}$

M images (epoch 0)



$$\mathcal{L}(\theta_i) = \text{MMD}^2(\text{True Data}, \text{Gen. Data}(\theta_i))$$

$$\theta_i \rightarrow \theta_{i+1}$$

QAG Model

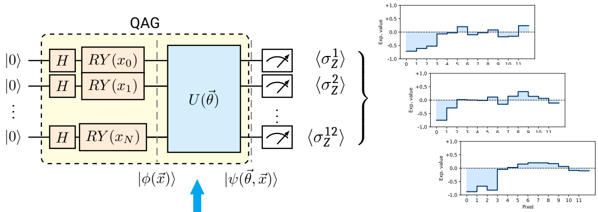
Training process

Choose a random initial set of parameters $\vec{\theta}_0$

For each epoch i

1. Generate M images
2. Evaluate the *MMD* loss
3. Update the parameters $\theta_i \to \theta_{i+1}$

M images (epoch 20)



$$\mathcal{L}(\theta_i) = \text{MMD}^2(\text{True Data}, \text{Gen. Data}(\theta_i))$$

$$\theta_i \to \theta_{i+1}$$

QAG Model

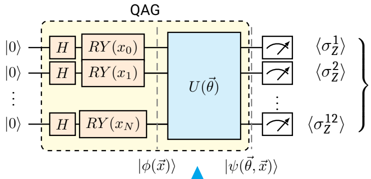
Training process

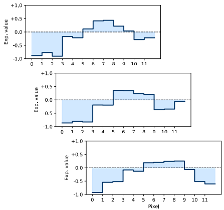
Choose a random initial set of parameters $\vec{\theta}_0$

For each epoch i

1. Generate M images
2. Evaluate the *MMD* loss
3. Update the parameters $\theta_i \rightarrow \theta_{i+1}$

M images (epoch 40)





$$\mathcal{L}(\theta_i) = \text{MMD}^2(\text{True Data}, \text{Gen. Data}(\theta_i))$$

$\theta_i \rightarrow \theta_{i+1}$

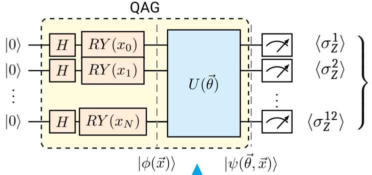
QAG Model
Training process

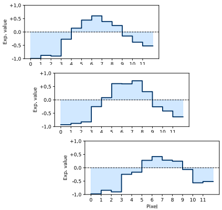
Choose a random initial set of parameters $\vec{\theta}_0$

For each epoch i

1. Generate M images
2. Evaluate the *MMD* loss
3. Update the parameters $\theta_i \rightarrow \theta_{i+1}$

M images (epoch 60)





$$\mathcal{L}(\theta_i) = \text{MMD}^2(\text{True Data}, \text{Gen. Data}(\theta_i))$$

$$\theta_i \rightarrow \theta_{i+1}$$

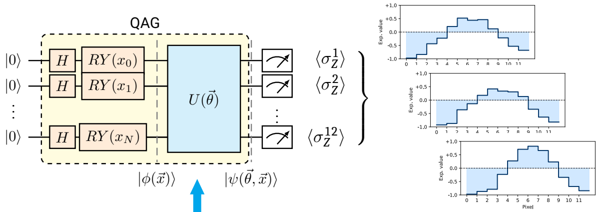
QAG Model
Training process

Choose a random initial set of parameters $\vec{\theta}_0$

For each epoch i

1. Generate M images
2. Evaluate the *MMD* loss
3. Update the parameters $\theta_i \to \theta_{i+1}$

M images (epoch 80)

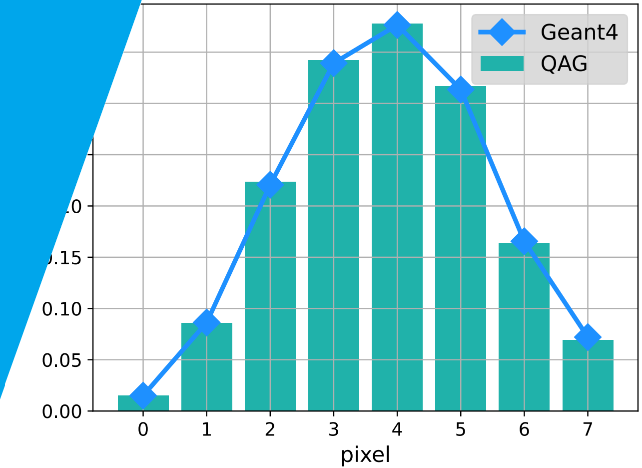


$$\mathcal{L}(\theta_i) = \text{MMD}^2(\text{True Data}, \text{Gen. Data}(\theta_i))$$

$$\theta_i \to \theta_{i+1}$$

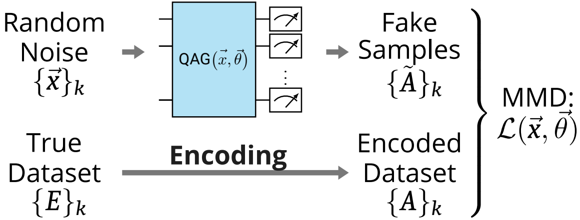
Model Evaluation

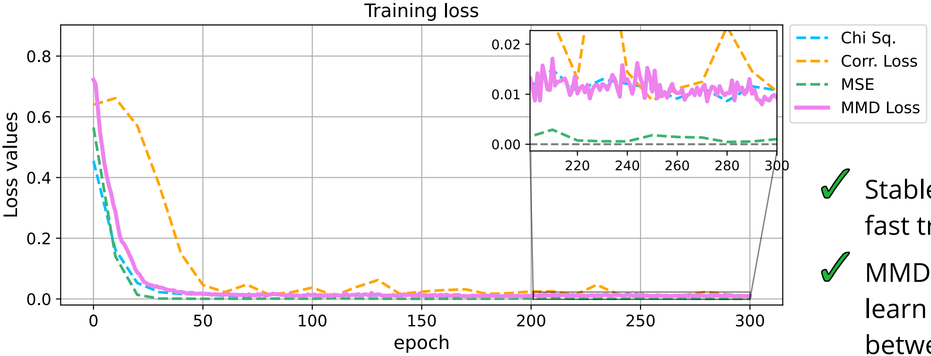
Overview of the model's performance on the EleScan dataset



Model Evaluation

Training



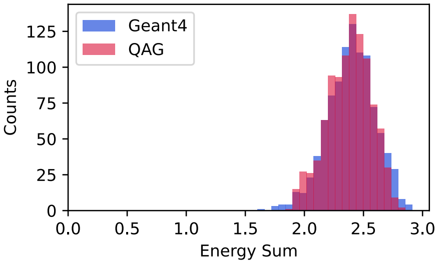


✓ Stable, smooth and fast training

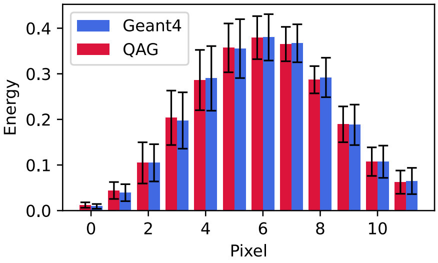
✓ MMD sufficient to learn correlations between pixels

Model Evaluation
Events generated

Total energy distributions of true (Geant4) and generated events



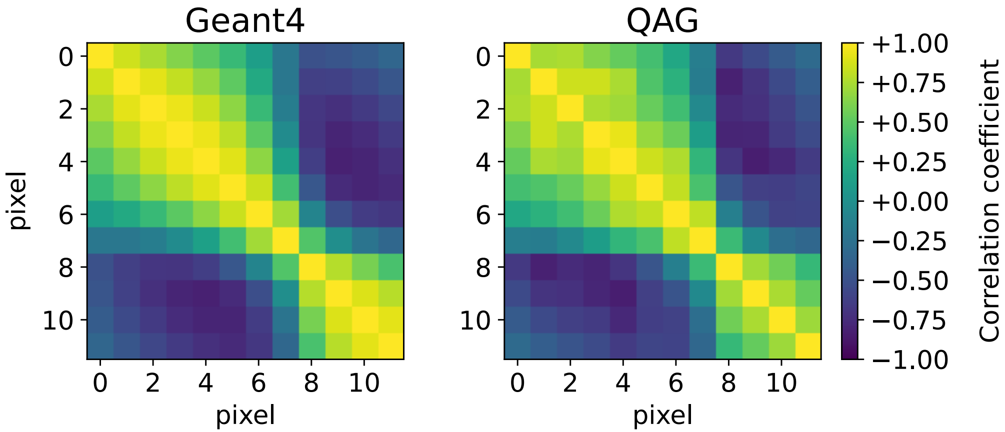
Average measured energy for each pixel



✓ High accuracy in both total energies and average pixel-wise energies

Model Evaluation

Correlations



✓ Model is able to reproduce correlations and anti-correlations of the showers

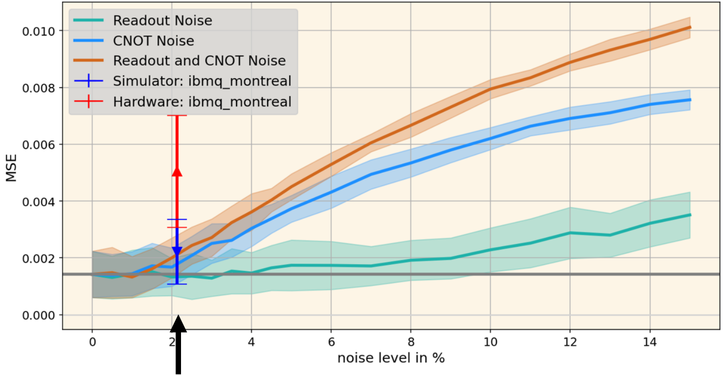
Model Evaluation

Noise study

Models trained without noise

- Models trained without noise
- Inference made under noise

⚠ Less accuracy in Hardware due to presence of swap gates in the transpiled circuit



Current noise levels on real hardware: $1-2\%$

$\rightarrow$ noise impact within the inherent uncertainty

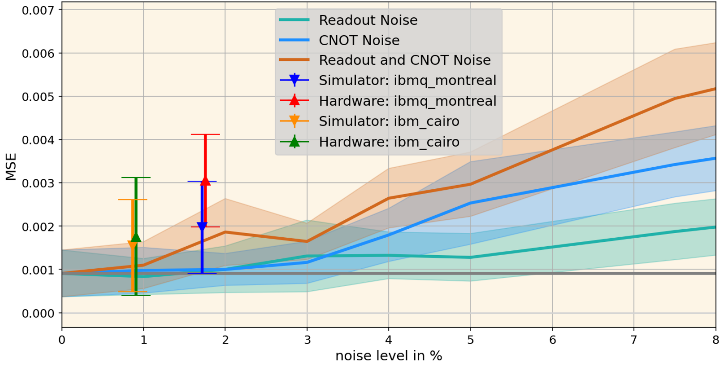
Model Evaluation
Noise study

Models trained in noisy instances

To be noted:
Different scale in x and y

- Models trained with noise
- Inference made with noise

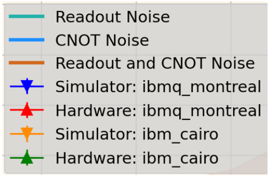
✓ Improved accuracy, the model is able to adapt its parameters to the noisy hardware to improve its precision



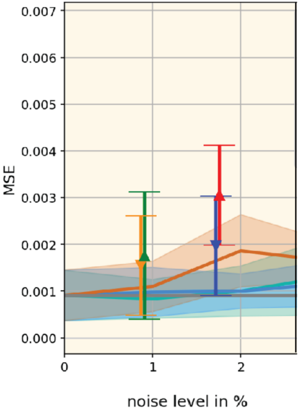
Model Evaluation

Noise study

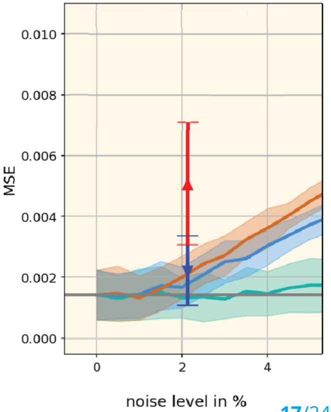
✓ **When trained directly on the noise instance** the QAG model is able to adapt its parameters to the noisy hardware to improve its precision

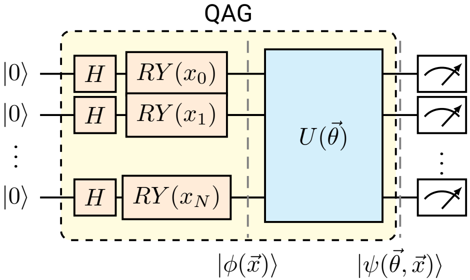


Models trained in noisy instances



Models trained in noiseless simulators





Conclusions

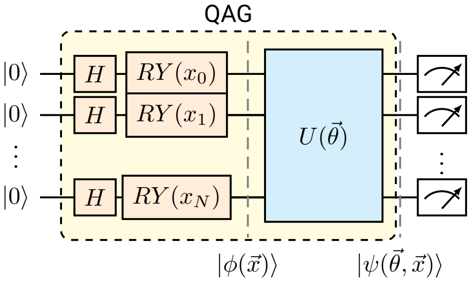
Summary of the model and future developments

Conclusions

Summary

QAG: a quantum generative model

- ✓ Efficient scaling with respect to gates and qubits
- ✓ Consistent, smooth, and rapid training convergence
- ✓ High inference accuracy
- ✓ Effectively adapts to noise in current NISQ devices



Future developments

- Conduct a more comprehensive hyperparameter optimization
- Overcome limit 1 qubit $\leftrightarrow$ 1 pixel
- Refine model geometry to align better to the specific physical problem

References

PhD Thesis:

Florian Rehm: ***Deep learning and quantum generative models for high energy physics calorimeter simulations***, RWTH Aachen University, PhD Dissertation (2023)

Paper:

Florian Rehm, Sofia Vallecorsa, Kerstin Borras, Dirk Krücker, Michele Grossi, Valle Varo: ***Precise Image Generation on Current Noisy Quantum Computing Devices***, Quantum Science and Technology (2023)

Questions?

Contact

Saverio Monaco

saverio.monaco@desy.de

github.com/SaverioMonaco/QAG (Private)

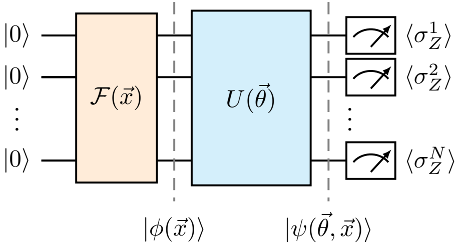
ENGAGE has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska-Curie Grant Agreement No. 101034267.

Backup slides

Backup slides

Types of generative models

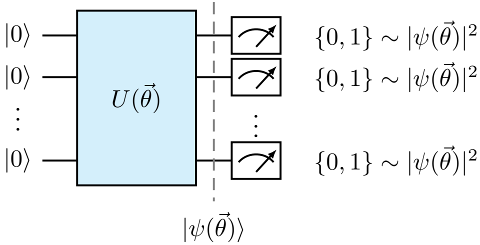
Continuous (QAG)



Source of entropy given by $\vec{x}$

- ✓ Fewer qubits needed
- ✗ Multiple measurement needed

Discrete (QCBM)



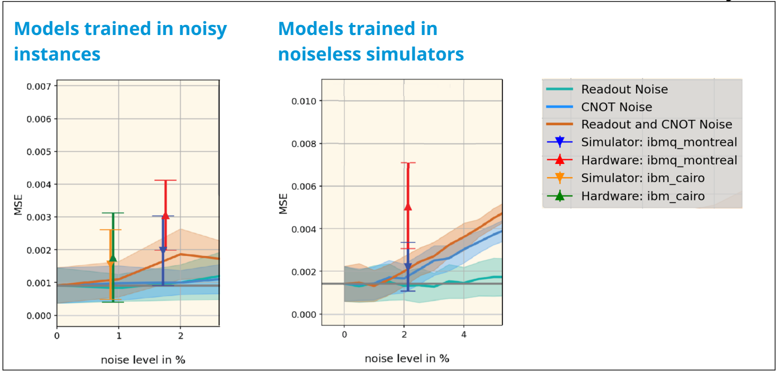
Source of entropy given by the measurement process

- ✗ More qubits needed
- ✓ Single measurement

Backup slides
Implementation on real hardware

- The noise analysis was conducted on **IBM superconducting** quantum computers.

Noise analysis:



Backup slides

Implementation on real hardware

- The noise analysis was conducted on **IBM** **superconducting** quantum computers.
- Additional tests are underway on newer and distinct architectures.

IQM — **Superconducting** quantum processor with a **star topology** and central resonator

parityqc — **Ion-trap** quantum processors

. . .